2003-03

An XML-based mission command language for autonomous underwater vehicles (AUVs)

Van Leuvan, Barbara C.

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL
Monterey California

THESIS

AN XML-BASED MISSION COMMAND LANGUAGE FOR AUTONOMOUS UNDERWATER VEHICLES (AUVs)

by

Darrin L. Hawkins
Barbara C. Van Leuvan

June 2003

Thesis Advisor: Don Brutzman
Co Advisor: Jeff Weekley

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2003	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE: Title (Mix case letters) An XML-based Mission Command Language for Autonomous Underwater Vehicles (AUVs)			**5. FUNDING NUMBERS**
6. AUTHORS Hawkins, Darrin L., Van Leuvan, Barbara C.			
7. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME AND ADDRESS Office of Naval Research			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			**12b. DISTRIBUTION CODE**

13. ABSTRACT

Autonomous Underwater Vehicles (AUVs) are now being introduced into the fleet to improve Mine Warfare capabilities. Several AUVs are under government-contracted development. Mission planning and data reporting vary between vehicles and systems. This variance does not pose an immediate problem, as only one AUV is currently in production. However, as more AUVs are put into production, commands will begin to get multiple AUVs. Without a single mission command language, multiple systems will require familiarity with multiple languages.

Extensible Markup Language (XML) and related technologies may be used to facilitate interoperability between dissimilar AUVs and extract and integrate mission data into Navy C4I systems. XML makes archive maintenance easier, XML documents can be accessed via an http server, and, in root form, XML is transferable on the fly by stylesheet.

This thesis presents an XML-based mission command for the command and control of AUVs. In addition, this thesis discusses XML technology and how XML is a viable means of achieving interoperability. Furthermore, this thesis provides an example mission file using existing software, and demonstrates the future of XML in AUV technology. Finally, this work ends with a compelling argument for the use of an XML-based mission command language to command all AUVs.

14. SUBJECT TERMS Autonomous Underwater Vehicles (AUVs), Tactical Command Language, telemetry validation, underwater robots, mine warfare			**15. NUMBER OF PAGES** 132
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

AN XML-BASED MISSION COMMAND LANGUAGE FOR AUTONOMOUS UNDERWATER VEHICLES (AUVs)

Darrin L. Hawkins
Captain, United States Air Force
B.S. Computer Science, Mississippi Valley State University, 1996

Barbara Van Leuvan
Ensign, United States Navy
B.S. Ocean Engineering, United States Naval Academy, 2002

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS TECHNOLOGY

from the

NAVAL POSTGRADUATE SCHOOL

June 2003

Authors: Darrin L. Hawkins
Barbara C. Van Leuvan

Approved by: Don Brutzman
Thesis Advisor

Jeff Weekley
Co-Advisor

Dan Boger
Chair, Department of Information Sciences

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Autonomous Underwater Vehicles (AUVs) are now being introduced into the fleet to improve Mine Warfare capabilities. Several AUVs are under government-contracted development. Mission planning and data reporting vary between vehicles and systems. This variation does not pose an immediate problem, as only one AUV is typically in operation at any given time. However, as more AUVs are put into production, cooperative operations will become possible and consistent mission commands will be necessary for multiple AUVs. Without a single mission command language, multiple systems will require familiarity with multiple languages.

Extensible Markup Language (XML) and related technologies may be used to facilitate interoperability between dissimilar AUVs and extract and integrate mission data into Navy C4I systems. XML makes archive maintenance easier, XML documents can be accessed via an http server, and, in root form, XML is transferable on the fly by stylesheet.

This thesis presents an XML-based mission command for the command and control of AUVs. In addition, this thesis discusses XML technology and how XML is a viable means of achieving interoperability. Furthermore, this thesis provides an example mission file using existing software, and demonstrates the future of XML in AUV technology. Finally, this work provides demonstration scripts and compelling arguments for the use of an XML-based mission command language to command all AUVs.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

We would like to express our sincere thanks to Dr. Don Brutzman, Doug Horner, Jeff Weekley, and Dr. Tony Healey for your motivation and support throughout this study.

We would also thank you for your guidance, wisdom, patience and enthusiasm throughout this project. Your efforts have made this an invaluable learning experience.

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

The U.S. Navy has been participating in Mine Warfare since the eighteenth century. While advancements have been made in mining, advancements in mine countermeasures had come to a stand still until the past several decades. While some improvement has been made, any classification and neutralization still usually involves the risk of an expensive piece of equipment, at best, or a human life, at worst.

Autonomous Underwater Vehicles (AUVs) are an increasingly popular solution to the problem of human involvement in mine hunting and countermeasures. Today, AUV technology is at a point that an individual AUV can be tasked to search for mines or collect bathymetry and environmental data. However, contracts for several different vehicles have been given to several different commercial companies. While each vehicle is intended for a separate area of operation, the probability that one command will one-day use several vehicles is strong. This is a problem because AUVs currently have no standardized mission planning language. As a result, multiple vehicles will require familiarity with multiple command languages and vehicles.

Most AUVs are commercially developed, and thus contain proprietary information. One of the biggest challenges facing the Navy's use of AUVs is the ability of the vehicle to communicate with others and to interface with Joint Command & Control, Communications, Computers and Intelligence (C^4I) systems. The lack of a common language creates a barrier between vehicles, and makes command and control of the AUVs more difficult

A solution to this interoperability problem is the use of the Extensible Markup Language (XML) and related technologies to create a mission command language. XML can be used to write logically, consistent and complete tasking orders. In addition, related technologies, such as Extensible Stylesheet Language for Transformations (XSLT) can be used to transform the XML-based tasking order into a text-based command file to task the AUV, or can be used to transform text-based outputs into a desired format for uploading into Joint C^4I Systems.

A preliminary set of XML elements and attributes describing a mission was developed based on the command language for the Naval Postgraduate School's (NPS) AUV. Some tags were chosen based on already existing tags in the DoD XML Registry. However, many of the necessary tags had not already been defined in other Namespaces. As a result, a proposed namespace specifically for AUVs was developed.

In addition to the tagset, a XML Schema Document was developed to validate mission-tasking orders. Finally, as an experimental test, an XSLT template was developed to transform an XML document into a text file to be inputted into the NPS AUV Workbench, a virtual simulation of AUV missions. Using XML and related technologies, generating virtually any type of data file is possible. Follow on work in this area includes the refinement of this language to be able to command all types of AUVs.

Another area of future work is to transform the collected data from any AUV mission into a form that is both validatable and machine readable, again using XML. This data can be incorporated into existing systems such as the Global Command and Control System, and such future concepts as the Semantic Web.

Finally, due to a hostile environment, underwater acoustic communications are fundamentally slow, insecure, and low-bandwidth. Other future work includes the development of binary compression, forward error correction, and XML digital signatures to work with AUV. Many of these issues have been addressed in other areas, and can be applied to underwater communications relatively easily.

I. INTRODUCTION

A. THESIS STATEMENT

Autonomous Underwater Vehicles (AUVs) currently have no standardized mission planning language and no uniform form for data output. The Extensible Markup Language (XML) and related technologies can be used to write logically, consistent and complete tasking orders. Data collection and metadata annotation can also be regulated. This approach will facilitate interoperability between dissimilar AUVs and extract and integrate mission data into Navy Command & Control, Communications, Computers and Intelligence (C^4I) systems.

B. OVERVIEW

AUVs are now being developed and introduced into the fleet to improve Mine Warfare capabilities. A family of diverse AUVs is being developed to accomplish this broad task. For these AUVs to be operationally effective, mission planning and data aggregation needs to be simple and transparent to the user. With such messaging support, one person can task and collect data from multiple vehicles without having to learn several different systems.

Several AUVs are under government-contracted development. Mission planning and data reporting vary between each vehicle and system. This does not pose an immediate problem for each AUV team, as only one AUV is in production, and only one subject matter expert (SME) is needed to run tasks on an AUV for a mission. However, as more AUVs are put into production, commands will begin to get more than one AUV. Until a means to control all of the AUVs with one language is developed, an SME will be needed for each type of AUV, leading to multiple SMEs within a command.

XML and Extensible Stylesheet Language for Transformation (XSLT) can be used to create a common mission planning and data formatting language for AUVs is a cost-effective means of achieving interoperability.

XML is a markup language and a World Wide Web (WWW) standard defined by the World Wide Web Consortium (W3C). XML is a markup language that provides structural information for documents. This structure defines the precise roles and relationships in which the information must follow within the document. A markup language defines the structure of a particular document. The XML specification defines a standard way to add markup to documents

(Walsh, 1998). XML differs from other markup languages because it does not directly specify how information is to be presented, but rather defines the structure (and thus semantics) of the information.

XSLT is a component of XSL (Extensible Stylesheet Language). XSL is a language for expressing style sheets. It consists of three parts: XSLT (a language for transforming XML documents), XPath (a language for defining parts of an XML document), and XSL Formatting Objects (a vocabulary for formatting XML documents).

Think of XSL as a language that can filter and sort XML data, a language that can define parts of an XML document, a language that can format XML data based on the data value, like displaying negative numbers in red, and a language that can output XML data to different devices, like screen, paper or voice. (www.w3schools.com/xsl/xsl_intro.asp accessed May 2003)

By using XML and XSLT, interested and even competing entities will be able to maintain their existing formats without adhering to an agreed upon standard. In addition to avoiding problems of adhering to a standard, the use of XML and XSLT can avoid disagreements in creating a standard.

C. OBJECTIVES

The objective of this thesis is to address the command and control (C^2) aspects of using XML to increase the utility of AUVs. XML programming will be addressed. Current mine warfare doctrine will be discussed only to introduce the topic and the need for this study. AUVs will also be introduced to clarify the need for a master control document. The operational limitations of existing AUVs will be discussed with regard to how these limitations affect C2, and also the future roles of AUVs and how a common vernacular could be helpful.

D. THESIS ORGANIZATION

Chapter II discusses the Navy's mine warfare doctrine, the current practices and the future of mine warfare. This chapter also examines the use of AUVs in mine warfare. Chapter III examines various AUVs, their uses, and their operational limitations. This chapter also examines the C2 aspects of these limitations. Chapter IV provides a brief history of XML, XSL and XSD, providing a detailed description of the W3C recommendations and what they are used for. Chapter V demonstrates a candidate vocabulary using XML to write a master mission-tasking document. XSL is then used to write style sheets that exchange data. This chapter will also

address the XML tagset necessary to write these documents. Chapter VI includes test runs of the process of going from XML Mission document thru the XSLT to outputs to the AUV. Chapter VII discusses the use of XML and XSL to exchange information between the GCCS MEDAL system and the AUV XML mission-tasking document. Chapter VIII addresses the impact of the Semantic Web on AUVs, and potential XML serialization for underwater communications. Data compression and security aspects of XML and related technologies are briefly addressed. Chapter IX discusses other work that needs to be done for unmanned undersea vehicle (UUV) common control station similar to unmanned aerial vehicle (UAV) station, UAVs, Divers, Marine mammals, Submarines, Explosive Ordnance Disposal (EOD) teams, and Mine Warfare Underwater Control Station.

THIS PAGE INTENTIONALLY LEFT BLANK

II. MINE WARFARE DOCTRINE

A. INTRODUCTION

The Navy and Marine Corps 'Forward…From the Sea' strategic concept has expanded naval operations into the littorals, an area where mines can be both a severe threat to the U.S. forces and a force multiplier against other forces. An effective Mine Warfare (MIW) force is necessary to ensure the Fleet's ability to carry out operations both in the open ocean and in the littorals. (After CSS Webpage, 2003)

B. MINE HISTORY

Early mines, developed by naval inventors such as David Bushnell and Robert Fulton, centered on the idea of striking ship's hulls with as explosive device. However, these keg mines were usually not in a stationary field, but were instead propelled by currents, harpoons, or underwater craft. These early mines were primitive, but inventors solved such problems as maintaining waterproof chambers for explosives and devising a trigger device. Until the second half of the nineteenth century, most major navies were not interested in mines, and saw them as weapons of states with weak navies. (After Mine Warfare History, 2003)

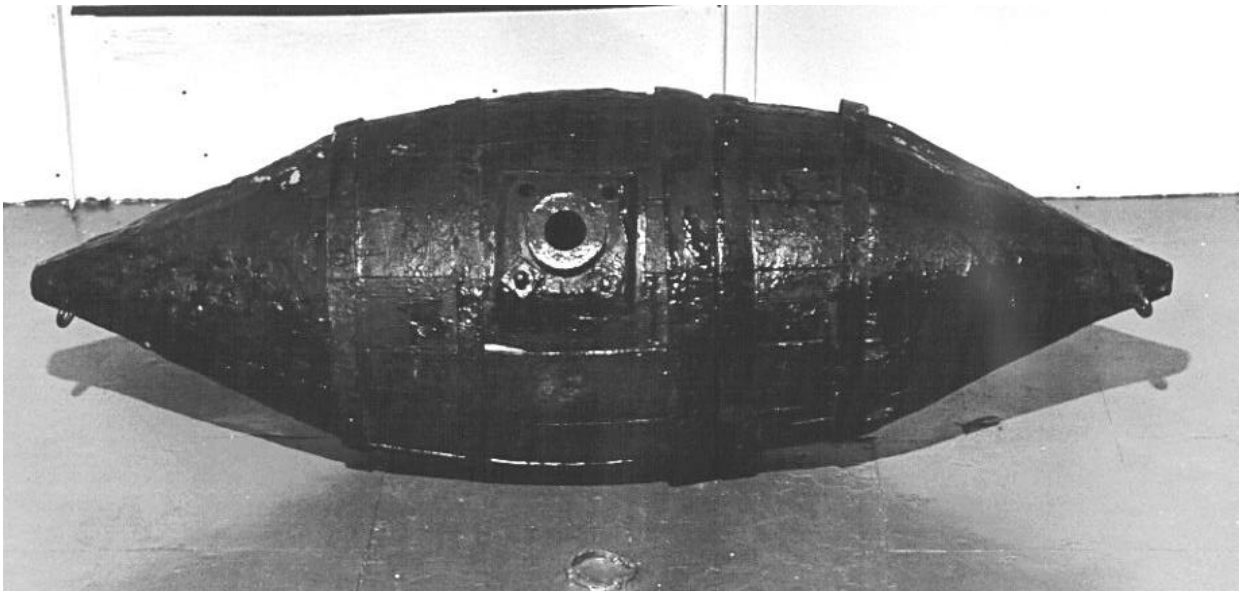

Figure 1. Bushnell Keg Mine (From The Bushnell Keg Mine, 2003)

During the second half of the nineteenth century, Russians in the Crimean War and the Confederacy used mines effectively during the Civil War. Mines became a common coastal

defense by the weaker naval powers forcing the U.S. Navy to investigate a variety of mine countermeasures. However, most mine countermeasures, while technically effective, were cumbersome, and no dramatic improvements from the previous fifty years occurred. (After Mine Warfare History, 2003)

By World War I, mines began to play a significant role in naval operations, and out of necessity, mine countermeasures became critical, especially to the Allies. Many advances in mine warfare occurred during WWI, and the United States acquired new skills and equipment needed to sweep some types of modern mines more effectively. However, in the years following WWI, the U.S. Navy made little progress in mine warfare, and in some areas, actually regressed, due to budget constraints, loss of experienced personnel, and lack of bureaucratic clout. (After Mine Warfare History (Web))

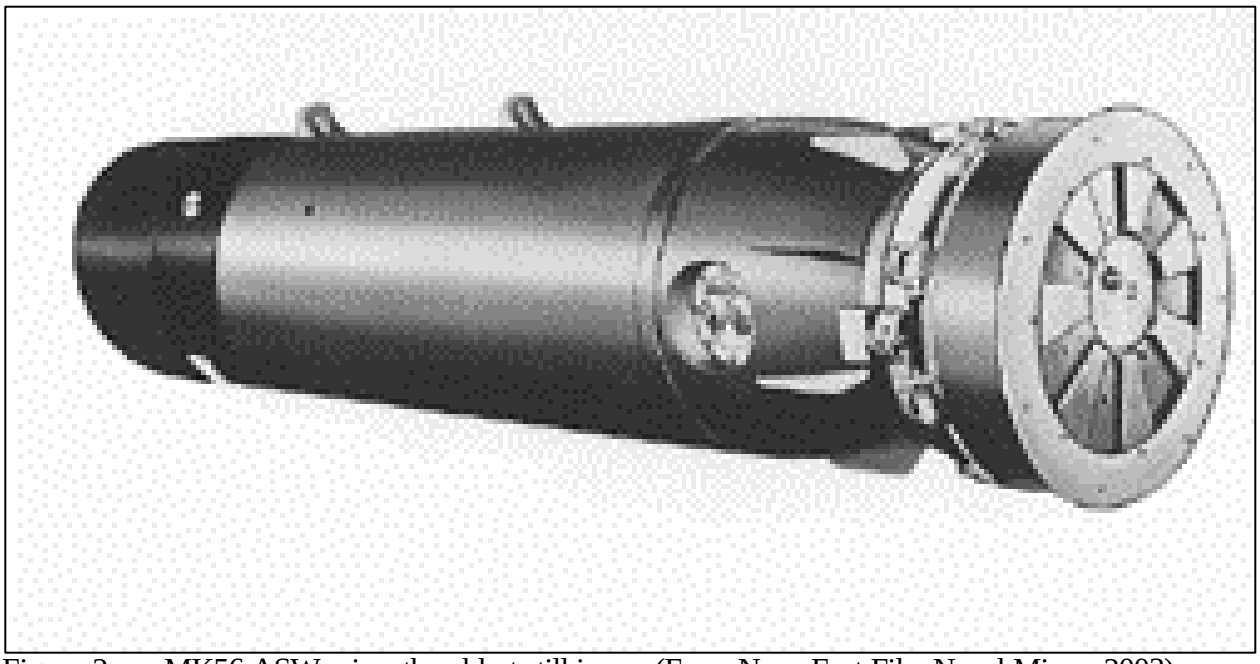

Figure 2. MK56 ASW mine, the oldest still in use (From Navy Fact File: Naval Mines, 2003)

As in WWI, mine warfare played a key role in World War II. By the end of the war, the United States had the world's largest minesweeping fleet, and had built up its own experience levels. WWII featured significant improvements in countermeasures, but the Allies were never completely successful in neutralizing the threat of mines. Throughout the Korean War, mines were easily one of the most dangerous weapons that the U.S. Navy faced. This threat caused renewed interest in mine countermeasures, which continued into the 1960s. However, in the early years of the Vietnam conflict, mines were not used in the same open ocean setting as

before. Mine countermeasures ships were required to operate in coastal areas, as part of a combined arms force. As a result, the Navy began emphasizing airborne mine countermeasures instead. (After Mine Warfare History, 2003)

Mines are relatively inexpensive, easy to procure, are difficult to track, and have a highly favorable return on investment. In addition, they are certain to play an important role in future engagements, especially in joint littoral warfare. Despite all of this, the United States Navy has devoted comparatively fewer resources to the development of mine warfare. (Mine Warfare History, 2003)

C. MINE WARFARE

Mine warfare (MIW) is defined as "the strategic and tactical use of sea mines and their countermeasures," (From Mine Warfare. NWP 3-15. Department of the Navy. August 1999. 1-2) and includes offensive, defensive and protective measures for laying and countering sea mines. Mine warfare can be broken into two distinct subdivisions – mining and mine countermeasures (MCM). Mining encompasses designing, producing, laying mines, while mine countermeasures covers the efforts of designing, producing and operating all forms of MCM equipment. (After Mine Warfare. 1-2)

D. MINING

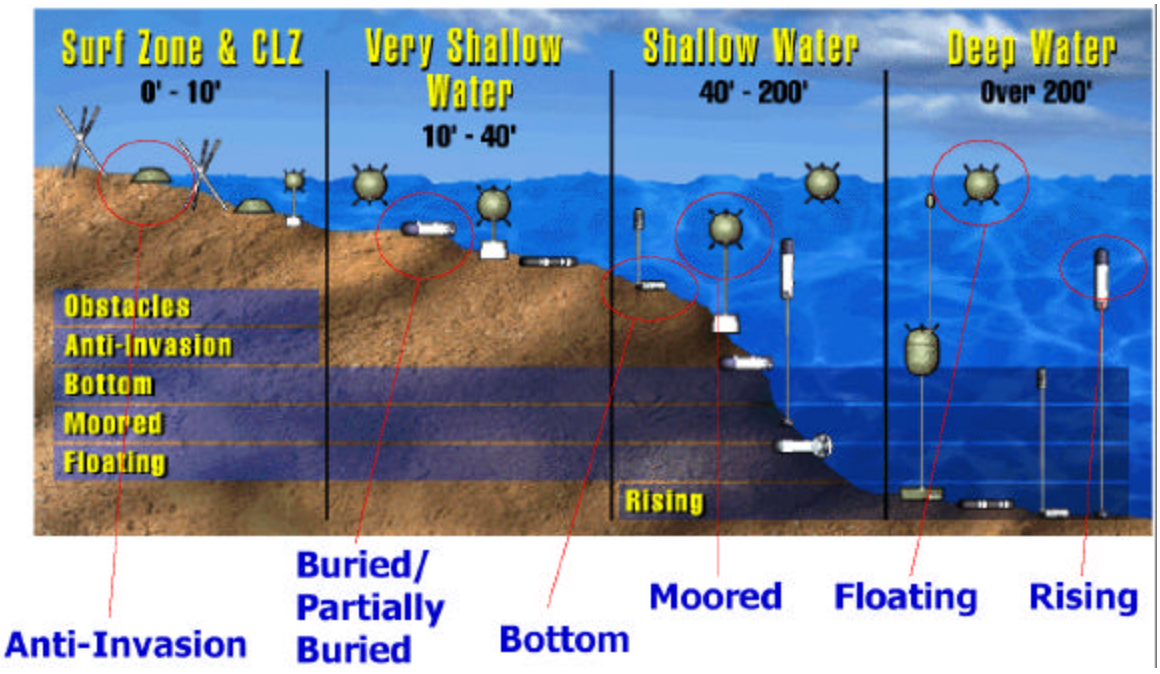

Figure 3. Mine Warfare Areas of Operation (From Marshall, Lehr, 1998)

For purposes of mine warfare, the littorals are broken down into several areas of operation. The area closest to the shoreline, called the surf zone or the coastal landing zone (CLZ), extends from water zero to ten feet deep. Typically, obstacles and anti-invasion mines are placed in this zone, as well as bottom, moored, and floating mines. The next zone is the very shallow water area, and covers water depths from 10 to 40 feet. The shallow water area covers 40 to 200 feet, and deep water extends to water greater than 200 feet deep. These last three areas typically contain buried or partially buried bottom mines, moored, floating, and rising mines. These areas can be seen in Figure 3.

Mining operations support the task of establishing and maintaining control of sea areas by using naval mines to inflict damage on enemy shipping and/or hinder, disrupt, and deny enemy sea operations. Mining operations have an advantage over other naval operations, because a minefield can inflict major long-term damage, without allowing for retaliatory action against the mine-laying force. In addition, a mine is armed 24 hours a day, from the time it is armed, until it is countered, or its useful life expires. (After Mine Warfare. 1-2)

Other advantages of mines include their covertness and surprise, their psychological effect on an enemy, and their ability to act as a force multiplier. In addition, the mine might be the only weapon that can apparently alter geography, as an area that has been mined must be avoided as if it were land. Finally, all of these advantages can be effective, even if the use of the mine is only simulated or threatened. The actual detonation of the mine might not be a significant factor in the effectiveness of the mines. (After Mine Warfare. 2-1)

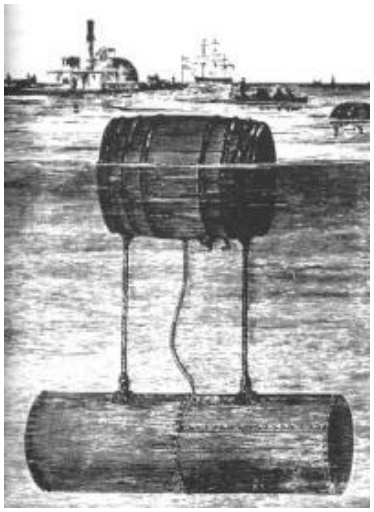

Figure 4. Confederate *torpedo* waiting for a target. (From Naval Mine History [AMCM])

A mine's passive nature produces most of its advantages, but also is its primary weakness. A mine must wait for a target and once laid, it does not discriminate. The stationary mine gives the target an opportunity to detect and then avoid or counter the minefield. Other disadvantages of mining include material degradation of the mine, and battery sensitivity to temperature. Another disadvantage of mining is the depth restrictions on where mines can be laid. (After Mine Warfare. 2-1)

E. MINE COUNTERMEASURES

Figure 5. USS AVENGER class Mine Countermeasures Ship (From USS AVENGER MCM 1)

MCM are classified as either defensive (enabling) or offensive (proactive). Offensive MCM are intended to prevent mines from being laid, and they eliminate the need for defensive MCM. Defensive MCM are classified as passive, preventing interaction between a mine and target, or active, which is reactive and involves interfacing directly with mines. (After Mine Warfare 3-1)

Offensive MCM intends to render ineffective one or more links in the mine laying process. Offensive MCM can be accomplished by destroying or disabling mines before they can be laid, destroying the enemy's mine laying capability, or mining to trap the enemy's ships in

port. Offensive MCM operations can be executed by strike or special operations forces, which have the capability of delivering an attack. (After Mine Warfare. 3-1)

Unlike offensive MCM, the objective of defensive MCM is to reduce the effectiveness of existing minefields. Defensive MCM is divided into active and passive MCM. Passive MCM can be divided into three categories. The first category is locating the threat through long-term intelligence collection, increased surveillance, and reconnaissance. After the threat is located, it must be localized. Localizing the threat means reducing the area in which shipping may be exposed to mines. The final category of passive MCM is reducing the risk. The primary means of reducing the risk for MCM forces are practicing precise navigation and influence signature control. (After Mine Warfare. 3-3)

Figure 6. USS RAVEN (MHC 61) Osprey Class. (From Navy Fact File: Coastal Mine Hunters, June 2003)

Active defensive MCM reduce the effectiveness of minefields by removing mines, destroying them in place, or neutralizing them. Active MCM includes mine hunting and minesweeping. Mine hunting is determining the location of mine in order to avoid, remove, render harmless, or destroy each mine. Mine hunting can be acoustic, magnetic, or optical; aircraft radar has also been used, but it has not produced dependable results. Minesweeping uses mechanical, magnetic, influence, or acoustic sweeps to cut the mooring cable of the mine or to actuate the mine. General MCM procedure is to mine hunt when environmental conditions permit and minesweep when mine hunting is not possible. This is because mine hunting in a

favorable environment is much safer for MCM assets than minesweeping. (After Mine Warfare. 3-5 – 3-7)

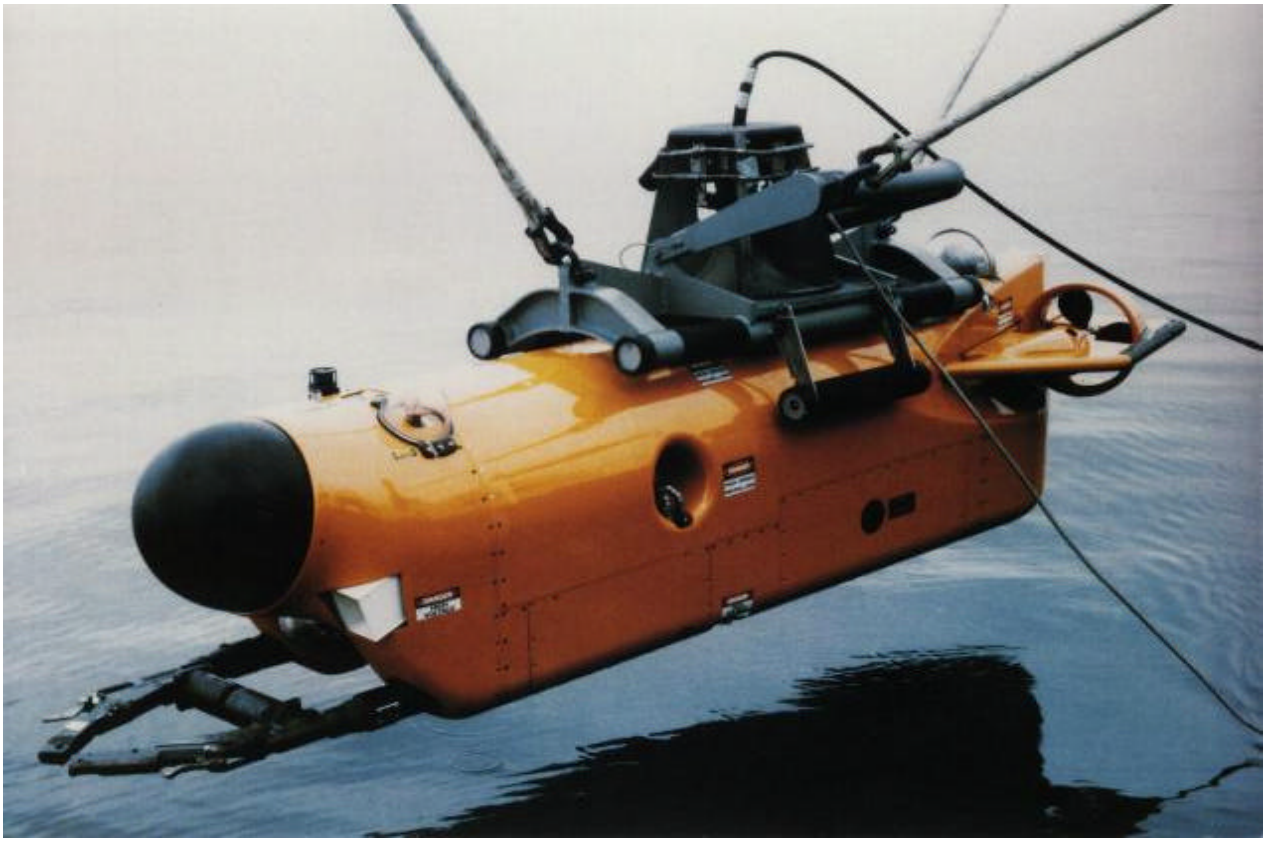

Figure 7. AN/SLQ-48 Mine Neutralization System (From AN/SLQ-48 Mine Neutralization System, 2003)

After a mine is located through mine hunting or minesweeping, it must be neutralized or countermined. One way to accomplish countermining is to use an explosive charge to cause the mine to detonate. A disadvantage of this is the requirement of a large explosive charge and/or closer placement to the mine, which may involve higher risk to the diver, ROV, or AUV. Alternatively, neutralization uses an explosive charge to damage the mine mechanism or rupture and flood the mine case. The major disadvantage of mine neutralization is that the mine case stays on the bottom without detonating the explosives. Other options include removal, which is relocation of a mine to an area where it poses no hazard and can be countermined or neutralized at a later time, or recovery, for the benefit of exploitation and analysis. (After Mine Warfare. 3-14 – 3-15)

F. SUMMARY

Mine warfare began in the 1700's with the use of 'torpedoes' that floated, waiting to be struck by a passing ship. Since then, advancements have been made in the mining area of mine warfare, but little has been done to improve mine countermeasures. Offensive mine countermeasures usually involve the destruction of a link in the enemy's mine-laying capabilities, while defensive mine countermeasures usually reduce the effectiveness of a minefield. While mine warfare has begun to improve in the last several decades, any type of mine neutralization still requires the involvement of either an expensive piece of equipment or an Explosive Ordnance Disposal (EOD) Officer.

III. AUTONOMOUS UNDERWATER VEHICLES (AUVs)

A. INTRODUCTION

Unmanned Underwater Vehicles (UUVs) are rapidly becoming a key player in the battlespace. With increasing littoral threats, autonomous underwater vehicles provide a capable option for meeting the Navy's needs. There are hundreds of UUVs and AUVs under development or commercially available, yet the fleet has little UUV based technology. Based on the pace of technology in the year 2000, a study team at Space and Naval Warfare Systems Center developed a vision of battlefield dominance via unmanned systems 50 years from now. Six years earlier, in the 1994 UUV Master Plan, four priorities were established:

1. "Near-term stopgap mine reconnaissance capability
2. Greatly improved, higher-performance mine reconnaissance capability
3. Surveillance, intelligence collection, and tactical oceanography capability
4. Research and development of enabling technologies for future UUV missions." (From Fletcher, 2000)

From this list of priorities, the Near-Term Mine Reconnaissance System and the Long-Term Mine Reconnaissance Systems evolved.

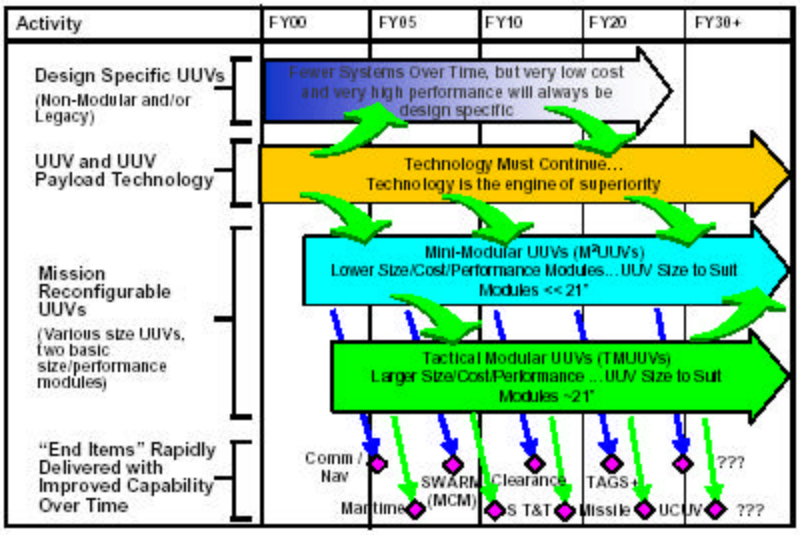

Figure 8. UUV Master Plan Summary Road Map (From Fletcher, 2000)

"Effective use of UUVs requires both appropriate technology and sound engineering." (From Fletcher, 2000) Technologies to be developed include autonomy, communications and sensors. The major focus of this thesis is on the development of increased communications among autonomous underwater vehicles (AUVs). Communications for any

single AUV or UUV is not a major problem, as the major concerns include available bandwidth, range, and covertness. However, communications among multiple vehicles operating together pose a much bigger challenge. (After Fletcher 2000) Currently, multiple AUVs are under development. Independently, each AUV is quite beneficial to the user, as a means of performing missions such as maritime reconnaissance, undersea search and survey, and communications/navigation aids to submarine track and trail. (After Whitman, 2002) Being able to task various AUVs using a common mission planning language greatly increases the benefit of AUVs.

B. LONG-TERM MINE RECONNAISSANCE SYSTEM (LMRS)

Figure 9. Long Term Mine Reconnaissance System (LMRS) (From Long Term Mine Reconnaissance System (Web))

The LMRS evolved from the second priority of the 1994 UUV Program Plan: "Greatly improved mine reconnaissance capability." (From Fletcher, 2000) In October 1999, a four-year development contract was awarded for initial operational capability in 2003. (After Fletcher, 2000) The LMRS is a UUV system capable of being launched clandestinely from a SSN688/688I or NSSN class submarine. (After Long–Term Mine Reconnaissance System (LMRS)) The system consists of a 21-inch diameter autonomous vehicle that would be stowed in the submarine's torpedo room, and could be launched and recovered through the torpedo tubes.(After LMRS ORD) Each vehicle was designed with forward and side-looking mine detection and classification sonar, as well as a homing/docking sonar. (After Long Term Mine Reconnaissance System)

C. REMOTE MINEHUNTING SYSTEM (RMS)

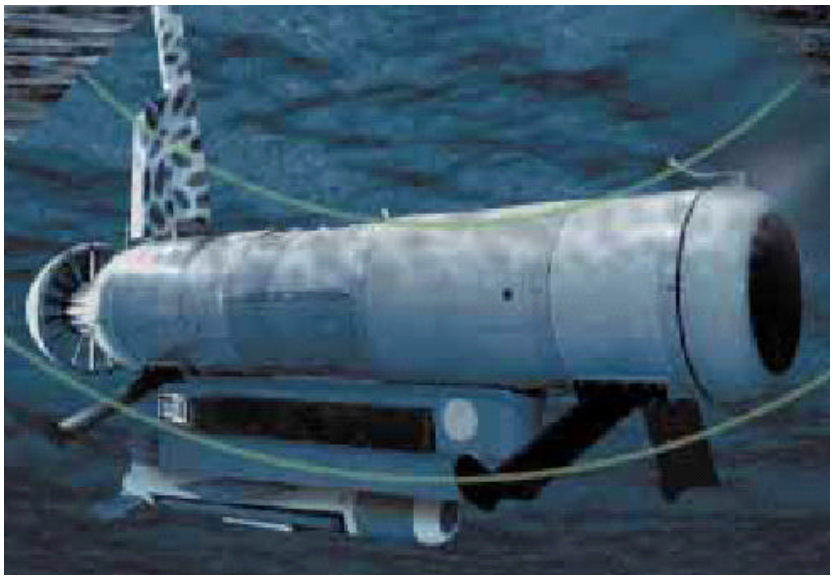

Figure 10. AN/WLD-1 Remote Minehunting System (RMS) (From RMS Brochure)

A second UUV is the Remote Minehunting System (RMS), "an organic, off-board system that will be launched, operated, and recovered from a host surface ship and will employ mine reconnaissance sensors." (From AN/WLD-1 Remote Minehunting System (Web)) RMS is intended for use in meeting the demand for over-the-horizon mine reconnaissance in support of an individual ship's mission, and also to prepare for the arrival of follow-on forces. The RMS will be used for reconnaissance against bottom and moored mines in deep water to a portion of the very shallow water.(From AN/WLD-1 Remote Minehunting System (Web)) The RMS will be operated and maintained by a surface ship, but will have self-contained command/control, propulsion, power, and navigational capabilities. (From AN/WLD-1 Remote Minehunting System (Web))

D. BATTLESPACE PREPARATION AUTONOMOUS UNDERWATER VEHICLE (BPAUV)

The BPAUV is a "small, fast underwater robot that maps the ocean bottom near the shore, detects changes in inshore conditions, and hints mines." (From NPS/CIRPAS Activity Statement, 2001) Because it is a larger unit, it is able to survey waters up to 300 meters deep, and conduct mine-hunting missions, and wide-area bottom mapping. (After Rose, 2002) However, because Bluefin Robotics Corp developed the BPAUV commercially, elements such as the source code are proprietary and may not be changed by any other company to provide new

functionality. (After Naval Coastal Sea Systems) This dilemma begins to introduce the need for creating a common mission and data formatting language for autonomous underwater vehicles using non-proprietary XML.

E. REMOTE ENVIRONMENTAL MONITORING UNITS (REMUS)

The REMUS is a low-cost AUV developed by and commercially available from the Oceanographic Systems Laboratory at Woods Hole Oceanographic Institute for coastal monitoring and multi-vehicle survey operations. (After WHOI at Sea) REMUS operates in depths from 10 to 66 feet and can be deployed by one or two men from a small craft without hoists. Another selling point of the REMUS is that data is downloadable into MEDAL format. (After Commerce Business Daily, 2001) Like the BPAUV, REMUS was developed commercially, and all source code is proprietary. Therefore, no other company can design and implement changes to the source code.

Figure 11. REMUS Variants "Darter," "Crevalle:" and "Gudgeon" (left to right) (Weekley, 2003)

F. AUV RESEARCH AT THE NAVAL POSTGRADUATE SCHOOL

The Naval Postgraduate School's (NPS) Center for AUV Research began in 1987, as a combined effort of the Departments of Mechanical Engineering, Computer Science, and Electrical and Computer Engineering. The Navy's interest in developing underwater vehicles for use in clandestine mine operations was very influential in the forming of the center. The Office of Naval Research sponsors most of the research performed by the Center, in collaboration with the Florida Atlantic University, with other funding coming from the National Science Foundation and the Naval Explosive Ordnance Disposal (EOD) Technical Head. The Center has

developed and tested three AUVs. The NPS AUV I and PHOENIX AUV are no longer in use by the Center, as the multi vehicle network server Acoustic Radio Information Exchange Server (ARIES) is currently operational in the Monterey Bay. In addition to ARIES, NPS has recently acquired the commercially built REMUS for other research.

The ARIES has been a test-bed for "development and evaluation of non-linear and adaptive control of vehicle motion. It has supported experimental work in system identification, and the development of high-speed graphics based physical modeling." (After NPS Center for AUV Research, June 2003) Numerous graduate and doctoral students have worked with ARIES as a communications server vehicle. The vehicle is also being used to develop low cost underwater navigation capabilities using commercial off the shelf (COTS) systems.

Figure 12. NPS ARIES on Deployment (From NPS Center for AUV Research, June 2003)

G. SUMMARY

Most existing and emerging AUVs are commercially developed, and thus contain proprietary information. One of the biggest challenges facing the Navy's use of AUVs is the ability of the vehicle to communicate with others and to interface with the GCCS-MEDAL system, which will be discussed further in Chapter 7. The lack of a common language creates a barrier between vehicles, and makes command and control of the AUVs more difficult. Restricting development by imposing a common language requirement is neither feasible nor cost-effective. At this point, XML appears to be the best option for creating a common mission and data scripting language for AUVs. By DoD directive, the use of XML must be non-proprietary, which would eliminate the need to always return to the developer when changes must be made.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. INTRODUCTION TO XML AND XSLT

A. INTRODUCTION

To begin a discussion about Extensible Markup Language (XML), one must talk about the World Wide Web Consortium (W3C). "The World Wide Web Consortium was created in October 1994 to lead the World Wide Web to its full potential by developing common protocols that promote its evolution and ensure its interoperability." (After W3C, 2003) It does this by developing technologies, specifications, guidelines, software, and tools that will create a forum for information, commerce, inspiration, independent thought, and collective understanding. The design goals for XML are shown in the table below.

1. XML shall be straightforwardly usable over the Internet.
2. XML shall support a wide variety of applications.
3. XML shall be compatible with SGML.
4. It shall be easy to write programs that process XML documents.
5. The number of optional features in XML is to be kept to the absolute minimum, ideally zero.
6. XML documents should be humanly legible and reasonably clear.
7. The XML design should be prepared quickly.
8. The design of XML shall be formal and concise.
9. XML documents shall be easy to create.
10. Terseness in XML markup is of minimal importance.

Table 1 XML Design Goals (After W3C, 2003)

Before XML, there was Standard Generalized Markup Language (SGML). "SGML is a meta language (i.e., a language for creating other languages) that is used to create markup languages, such as Hypertext Markup Language (HTML)." (From Deitel, Deitel, Nieto, Lin, and Sadhu, 2001)

B. EXTENSIBLE MARKUP LANGUAGE (XML)

A Working Group of twelve professionals with significant shared experience, both with the World Wide Web and with using computers to process and manage information using SGML came together. The Web was growing exponentially and they wanted to use it to publish their SGML-encoded information. Although SGML was a ten-year-old technology, it was very powerful and it made information reusable. It had the ability to describe information in a way that was system-independent. But SGML had its problems; it was difficult to learn, its acceptance was limited to documentation professionals, and it was very difficult to use SGML with this new medium known as the Web. The group formed around the idea that the two technologies could be made to work together to make it easier to share and reuse information. (After Berners-Lee with Fischetti, 1999)

Working under the auspices of the W3C, they embarked on a journey to create this nameless subset of SGML which would make it easy to use on the Internet, support a wide variety of applications, be compatible with SGML, and, ultimately, change the world. The goal of bringing together the two powerful ideas of the Web and of descriptive markup energized the group and drove them to work evenings and meeting by teleconference. Whenever we lost our way, someone would ask, “Is this feature necessary for success?” The group worked to transform these goals and experiences into a formal language, a language designed to make sharing reusable information ubiquitous. This became known as XML.

```
<scene>
  <FX>General Road Building noises.</FX>
  <speech speaker="Prosser">
    Come off it Mr Dent, you can't win
    you know. There's no point in lying
    down in the path of progress.
  </speech>
  <speech speaker="Arthur">
    I've gone off the idea of progress.
    It's overrated
  </speech>
</scene>
```

Figure 13. Sample XML file (From ‘What is XSL?’ 2003)

The group knew SGML was the best approach for reusing the kinds of information they worked with, but they needed to make SGML easier to learn, understand, and implement, while retaining its core values, “SGML fit for the Web.” The core value of SGML that they wanted to

build into XML was that of “descriptive markup." “Markup is information inserted into a document that computers use; in the case of SGML, markup takes the form of tags inserted into documents to mark their structure. Descriptive markup uses markup to label the structure and other properties of information in a way that is independent of both the system it's created on and of the processing to be performed on it.” (From Hollander and Sperberg-McQueen 2003)

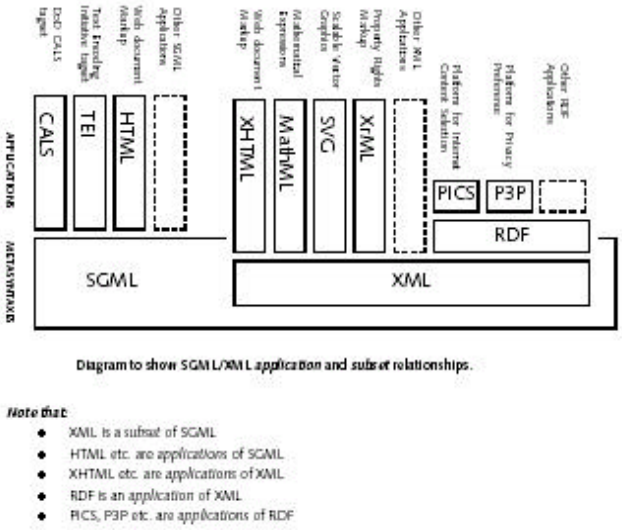

Diagram to show SGML/XML *application* and *subset* relationships.

Note that:

- XML is a subset of SGML
- HTML etc. are *applications* of SGML
- XHTML etc. are *applications* of XML
- RDF is an *application* of XML
- PICS, P3P etc. are *applications* of RDF

Figure 14. SGML – XML Relationship (From Just what is XML? June 2003)

They wanted XML, like SGML, to be a meta-language. They wanted it to create vocabularies that is relevant to their information and to enable user-defined, processing-independent markup that are easier to reuse and can be processed in new and often unexpected ways.

SGML, fit for the Web, would make it easy and reliable for computers (and humans) to use descriptive, structural markup in their documents. By using descriptive data tags, the information owner can make documents into semantically rich data and avoid what they called “crufty tag salad”, presentation-oriented markup used just because it looks right.

The result was a 25-page XML specification that could be easily learned and implemented. XML, a meta-language that allows design markup languages that describes what is important to the user. It provides elements and attributes to capture logical structure and enables semantic understanding. They were able to balance features against complexity. Their practical litmus test “is it necessary for success?” helped us create a language fit for the Web.

A data object is an XML document if it is well formed, and may also be valid if it meets certain constraints. Well-formed documents do not have to be created in a structured environment, against a pre-defined set of structural rules, but merely have to comply with XML well-formedness constraints. Well-formed XML elements are defined by their use, allowing authors to tailor elements to their development. This flexibility gives authors greater control over document processing and design. This is a great improvement over traditional SGML environments, in which structure must be formally defined before any documents can be written.

C. XML SCHEMAS

A schema is a set of rules that a document follows, which software may need to read before processing and displaying a document. Valid XML differs form well formed XML in its relationship to a schema. Well-formed XML is designed for use without a schema, whereas valid XML explicitly requires it. Table 1 lists everything that a schema can be used to define.

1. Elements that can appear in a document.
2. Attributes that can appear in a document.
3. Which elements are child elements.
4. The order of child elements.
5. The number of child elements.
6. Whether an element is empty or can include text.
7. Data types for elements and attributes.
8. Default and fixed values for elements and attributes.

Table 2 Schema Definitions (After Introduction to XML Schema)

Once written, a schema allows the user to check whether an XML document is valid. Valid XML documents employ features that can significantly improve the usability of a document, including: linking mechanisms, entities and attributes. Most XML Web sites are likely to be composed of valid XML documents. Using a schema gives creators the freedom to structure their sites and use much greater feature sets than HTML has traditionally allowed. The process for validating a schema is shown in Figure 15.

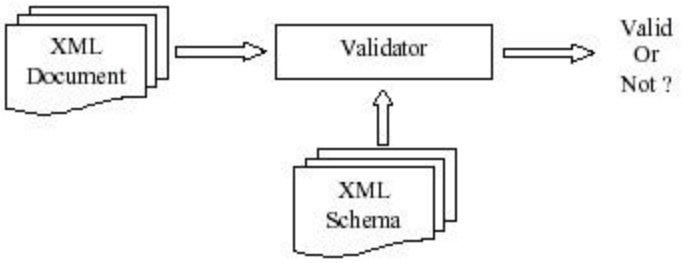

Figure 15. XML Schema Validation Process (From Serin, 2003)

"Document authoring, processing, storage and display are made easier because documents exist in a structured environment. Authors must create documents against a pre-defined structure and benefit from a clear document model. Like well-formed XML, valid documents must be accompanied by stylesheets to achieve visual display." (From Valid XML**)**

The original group of twelve with its common, shared experience gave way to lots of groups with differing goals and backgrounds. XML grew stronger for the new insights. You now have XML + XLINK + XSL + Namespaces + Infoset + XML Linking + XPointer Framework + XPointer namespaces + XPointer xptr() + XSLT + XPath + XSL FO + DOM + Sax + stylesheet linking PI + XML Schema + XQuery + XML Encryption + XML Canonicalization + XML Signature + DOM Level 2 + DOM Level 3.

D. EXTENSIBLE STYLESHEET LANGUAGE FOR TRANSFORMATIONS (XSLT)

One of the tools mentioned above, which is pertinent to this thesis, is Extensible Stylesheet Language (XSL). XSL is a language for expressing style sheets. It is made up of three components:

1. XSL Transformations (XSLT), a language for transforming XML documents
2. XML Path Language (XPath), an expression language used by XSLT to access or refer to parts of an XML document
3. XSL Formatting Objects (XSL-FO), an XML vocabulary for specifying formatting semantics

An XSL style sheet is, like with Cascading Style Sheets (CSS), a file that describes how to display an XML document of a given type. XSL shares the functionality and is compatible with Cascading Style Sheets, level 2 (CSS2), a style sheet language that allows authors and users to attach style (e.g., fonts, spacing, and aural cues) to structured documents (e.g., HTML documents and XML applications), although it uses a different syntax. It also adds:

- A transformation language for XML documents: XSLT. Originally intended to perform complex styling operations, like the generation of tables of contents and indexes, it is now used as a general purpose XML processing language. XSLT is thus widely used for purposes other than XSL, like generating HTML web pages from XML data.
- Advanced styling features, expressed by an XML document type which defines a set of elements called Formatting Objects, and attributes (in part borrowed from CSS2 properties and adding more complex ones (After The Extensible Stylesheet Language)

Styling requires a source XML documents and a style sheet. The source document contains the information the style sheet will display while the style sheet describes how to display a document of a given type. Figures xml1, xml2, and xml3 show a sample XML file, two sample templates from a style sheet and the rendering of them, respectively.

Separating the source document's content and its styling information allows displaying the same document on different media (like screen, paper, cell phone), and it also enables users to view the document according to their preferences and abilities, just by modifying the style sheet.

```
...
<xsl:template match="FX">
  <fo:block font-weight="bold">
    <xsl:apply-templates/>
  </fo:block>
</xsl:template>

<xsl:template match="speech[@speaker='Arthur']">
  <fo:block background-color="blue">
    <xsl:value-of select="@speaker"/>:
    <xsl:apply-templates/>
  </fo:block>
</xsl:template>
...
```

Figure 16. Sample XSLT (From What is XSL?)

The style sheet can be used to transform any instance of the schema/Document Type Declaration (DTD) it was designed for. “The first rule says that an FX element will be transformed into a block with a bold font. <xsl:apply-templates/> is a recursive call to the template rules for the contents of the current element. The second template applies to all speech elements that have the speaker attribute set to Arthur, and formats them as blue blocks within

which the value speaker attribute is added before the text. (After The Extensible Stylesheet Language)

General Road Building noises.

Prosser: Come off it Mr. Dent, you can't win you know. There's no point in lying down in the path of progress.

Arthur: I've gone off the idea of progress. It's overrated

Figure 17. Sample Output (From What is XSL?)

The above rendering is the Formatting Objects (XSL-FO) generated by the XML file and two sample templates from a style sheet. The XSL-FO vocabulary is designed to allow information to be displayed on a wide variety of media: screen, paper, or even voice.

E. SUMMARY

In this chapter XML and XSLT are discussed. The roots of XML are given, XSLT is briefly expounded on, and some other related technologies are mentioned to provide the background and basis upon which this common data and formatting language will be built.

THIS PAGE INTENTIONALLY LEFT BLANK

V. USING XML AND XSLT TO INCREASE AUV INTEROPERABILITY

A. INTRODUCTION

Twenty years ago, software only worked with other software bought from the same vendor. Today, consumers rightly expect software components to be interchangeable. The W3C, a vendor-neutral organization promotes interoperability by designing non-proprietary computer languages and protocols to avoid the market fragmentation of the past. (From W3C in 7 Points)

B. XML AND INTEROPERABILITY

As stated in Chapter III, at least four different AUVs either exist or are under development: one for each area of operation. Hydroid, Inc. manufactures REMUS (After Hydroid, Inc), Bluefin Robotics manufactures BPAUV (After ONR BPAUV). Lockheed Martin has the contract for RMS and Boeing has LMRS. Interoperability is defined as the ability of systems, units, or forces to provide services to and accept services from other systems, units, or forces and to use the services so exchanged to enable them to operate effectively together. (After Defense Technical Information Center) Currently, most AUVs are commercially developed, contain proprietary internal architectures, and, thus display poor interoperability.

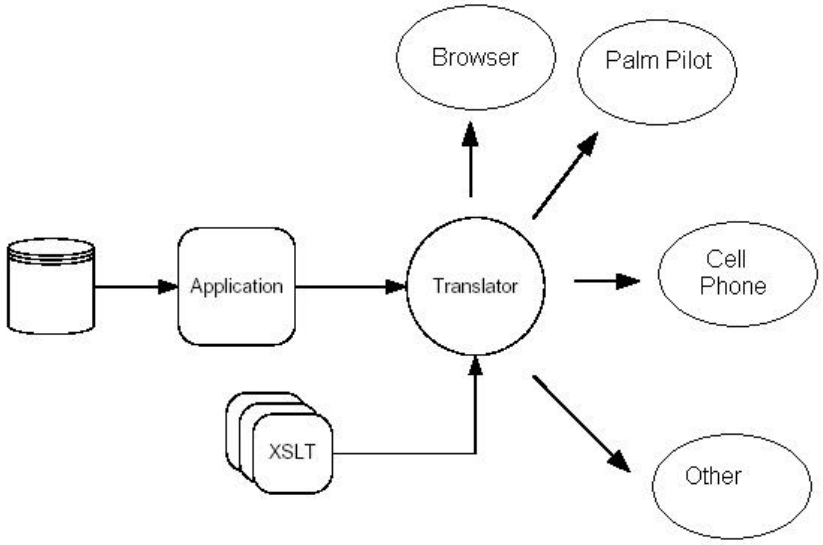

Figure 18. XML Interoperability (After Wrox Diagram)

One method of improving interoperability is through the use of a common XML-based mission planning and data formatting language. Similar XML-based languages have already

been used in various applications to achieve software interoperability. For example, the Schools Interoperability Framework, a division of the Software and Information Industry Association demonstrated interoperable applications in November 2001. The software interoperability was exhibited in a network environment, in which data is shared between applications through a "series of standard messages, queries, and events written in XML." (From Schools Interoperability Framework, June 2003)

Numerous other examples of the use of XML to increase software interoperability exist. The IMS Global Learning Consortium uses XML binding because of its ease of maintainability and increased flexibility. (After IMS Global Learning Consortium) The Open Travel Alliance (OTA) formed to "improve the electronic exchange of business information across all sectors of the travel industry."(From OTA XML Specification, June 2003) To assist programmers with the implementation of a cross-industry effort to improve this exchange, OTA released a Specification Document, a schema and schema fragments, a Document Type Definition (DTD), a Universal Modeling Language (UML) Model, a Data Dictionary, and Appendices. (After OTA XML Specification, June 2003) The retail industry has joined the XML drive, with the International XML Retail Cooperative (After IXRetail), which is intended to ease application-to-application integration within a retail enterprise. The IXRetail initiative began in 2000, when the Limited began looking at XML for application integration. The Limited concluded that XML could be used for application-to-application integration within the enterprise, application construction and evolution of the Enterprise Application Integration (EAI).(After ARTS/IXRetail XML Event, June 2003) Finally, the health care industry is joining the push towards the use of XML, as well, with the HL7/XML Interface. Health Level 7 (HL7) is "the dominant health care standards organization for healthcare message communication from machine to machine in the United States, with an active presence in Europe, Australia, and most recently in Japan." HL7 used XML for both healthcare messages and clinical record documents, and represented the culmination of three years of work in bringing together the HL7 communication protocol with the XML markup strategy. (After Hl7-XML Progress Report, June 2003)

As evidenced by the above examples, XML is becoming a common answer to the interoperability problems faced by any industry. This is because XML is designed with the ability to describe information in a way that is system-independent. In addition to being relatively simple to write, a user can easily export an XML document to both XML and non-

XML formats. XML also makes archive maintenance easier, as will be addressed in a later chapter. XML documents can be accessed via an http server. Finally, in root form, XML is transferable on the fly by stylesheet. Thus XML and related technologies can be used to improve interoperability between AUVs.

C. CONSTRUCTING THE MISSION COMMAND LANGUAGE

The first step in constructing a Common XML-Based Mission and Data Formatting Language using XML is defining a tag set. A tag set is the set elements and attributes use to describe what is trying to communicated or described. To make a language common and interoperable, tags must come from a central XML registry that allows common access. XML registries are a vital component in the implementation of shared data exchanges. The DoD XML Registry constitutes guidance in the generation and use of XML among DoD communities of interest and is the authoritative source for registered XML data and metadata components.

Researching the DoD XML Registry resulted in the realization that only some of the necessary tags for a common mission and data formatting language exist. One output of this thesis is a proposed AUV Namespace. (See Appendix E) "Namespaces are technical mechanisms that allows overlapping XML to be tagged with distinguishing labels." (From DoD Metadata Registry and Clearinghouse) Namespaces make up collections of data constructs that share a common context within a Community of Interest (COI) that can be leveraged for XML administrative purposes. A COI is a group of people, agencies, activities, and system builders who share an interest in a specific domain.

After defining the tagset, the XML schema document can be written. The schema document is the modeling document, which defines the structure of the input XML documents. The schema is used to validate these documents and uses the same syntax that XML uses; while fully supporting the Namespace Recommendation. In addition, the schema allows creation of complex and reusable content models with the idea of object inheritance and type substitution. The fundamental idea behind validation is to create XML documents that they can be shared by multiple users without any conflict when they follow the same rules that the schema defines. Any well-formed XML document can be validated against any schema. (After Serin, 2003). Using the schema document as a guide, an XSLT is written to transform some input into a user-defined output. Examples of those outputs are illustrated in the next chapter.

D. TRANSFORMING THE DOCUMENT

XSLT stylesheets use XML Path Language to match nodes when transforming an XML document. Xpath provides syntax for locating specific parts of an XML document. Once Xpath has located and matched a node, an XSLT Processor can transform the document into another form, whether it is XML, HTML, plain text, or any other text-based document. (Deitel, Deitel, Neito, Lin, and Sadhu, 2001) One benefit of using XSLT for such a task is that XSLT has none of the 'side effects' of correct order and code running the way it is written. XSLT, on the other hand, will run correctly, regardless of the order in which tasks are performed. In addition, an XSLT engine can run the code in a stylesheet in any order. Because XLST is a declarative language, rather than an imperative one, it can even run multiple pieces of code simultaneously, effectively optimizing the program. A visual representation of the stylesheet function is shown below.

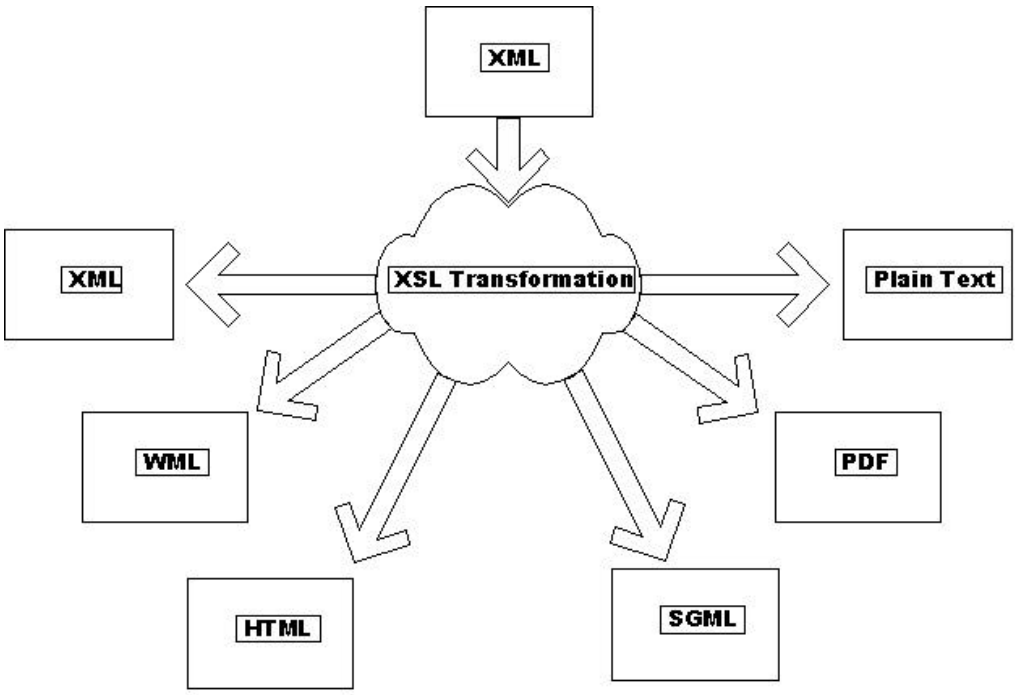

Figure 19. Demonstration of XSLT Functions (From XML – An Introduction, June 2003)

E. ARCHIVING XML DATA

Digital publication preservation is a significant part of tomorrow's heritage. Without a concerted effort, the digital information of today will not be available. By correctly archiving data, especially the data collected from AUV missions, MEDAL, and other similar databases

continue to be able to access this data. XML documents can be archived wither of two places, on the file system itself or in a database. While the file system is fast and simple, it is not really practical for large applications, so the logical choice is to archive data in a database. There are two types of databases. First there is a Relational Database Management System (RDBMS), which offers many advantages such as tools for data mining, but at the expense of needing transformations into a relational data model and translations for queries. Another option is the use of Native XML databases, which have much better performance for storage, retrieval, and query, but lack the advantages of a mature RDBMS product.

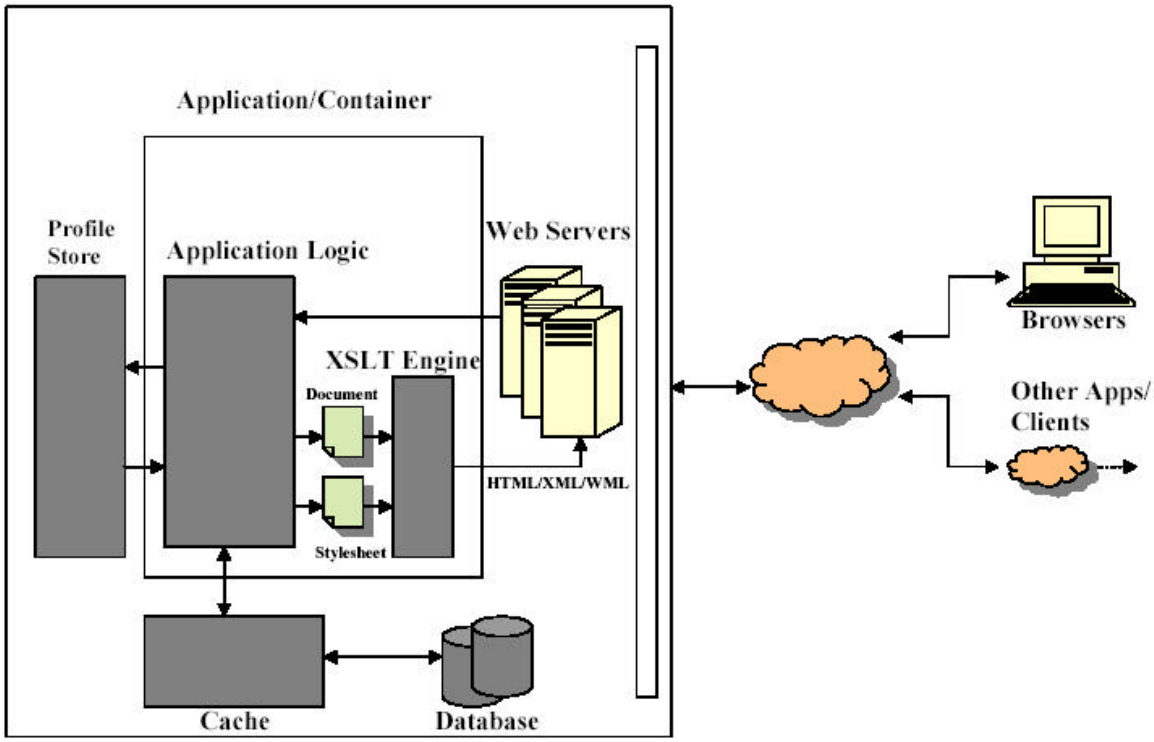

Figure 20. XML Archiving Process (From Ipedo Web, June 2003)

F. SUMMARY

This chapter discussed the use of XML and related technologies for increasing the interoperability of software in various industries, including retail, healthcare, and online learning. Throughout the development of this thesis, the DoD XML Registry was searched for applicable tags to define an AUV Namespace. Because most of the necessary tags are not currently in use, a proposed AUV Namespace was developed, and can be seen in Appendix F. After developing a

tagset, a schema document was developed to validate the mission documents. Finally, as a experimental test, an XSLT template was developed to transform the XML mission document into a text-based file.

VI. AUV SIMULATION WORKBENCH

A. INTRODUCTION

This chapter describes the AUV Workbench and gives a sample mission document. An AUV Workbench mission script file is presented first. This script file is incorporated into the mission command language to show the capability to output data that can be use by the AUV Workbench.

B. AUV WORKBENCH OVERVIEW

The AUV Workbench is used by the scripts developers and by the thesis students to test and edit their simulations. It will also allow for the pre- visualization of in-water missions. It is designed to:

- Simplify and make more easily the utilization of the simulation
- Have all the windows in one main window
- Allow scripts developers to edit and test their scripts more quickly
- Allow AUV software developers to evaluate execution and dynamics improvements

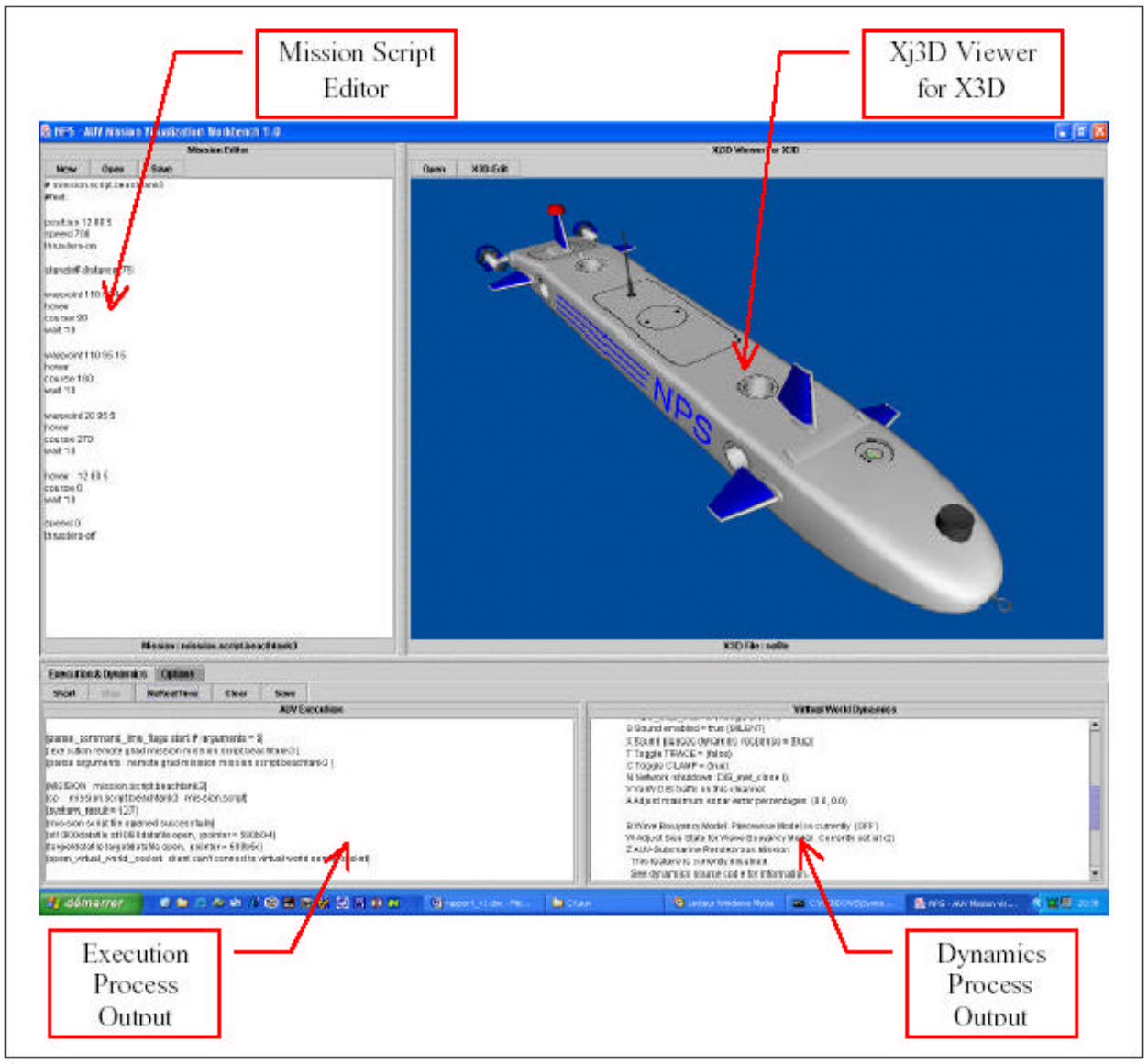

Figure 21. Interface of AUV Workbench (Gruneisen and Henriet, 2002)

The Mission Script Editor is a text editor with which users can create, open or save missions. When a mission is opened, the software automatically creates a backup of this file (with the name of the mission and the date). So, the user can reopen this file in case of a modification error. The mission opened will be the mission used for the simulation.

The Execution and Dynamics panels allow the user to launch or stop the simulation of the Mission File in the Mission Editor. They allow the user to display the simulation in real-time or not, clear the execution and dynamics text area and save the execution and dynamics text area in two different files, MissionName_Execution_Date and MissionName_Dynamics_Date, respectively.

The Options Panel allows the user to select Execution Program: there are two different programs for the execution level (one in C and the other in Java), so the users can select which one to use; and select AUV Model: there are four different AUVs which each have different dynamics coefficients, so the users can select which coefficients they want to use for the simulation.

The simulation process works be means of the Java language that allows developers to execute others programs from a Java program. Here, there are two different Threads (one for Execution and one for Dynamics) which have launch the programs, catch the output streams, and print them in the two text areas in the main window.

The Extensible Java 3D Graphics (Xj3D) application programming interface (API) and veiwer uses all the specification of Extensible 3D Graphics (X3D) to be able to display a Virtual Reality Modeling Language (VRML) file in a program using Java3D. "However, Xj3D is currently under development, so not all of the X3D nodes are integrated in Xj3D (Billboard for example); thus, it is necessary for the users to download and install the latest version of the Xj3D package to update the Java classes which are used by the program." (After Gruneisen and Henriet, 2002)

C. DESCRIPTION OF USE OF TAGSET AND SCHEMA IN CONJUNCTION WITH THE AUV WORKBENCH

Below is an inclusive tag set based on the AUV mission script help file and the AUV Workbench.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <!-- Sample XML file generated by XMLSPY v5 U (http://www.xmlspy.com)-->
-         <AUVMission         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="C:\Documents                                    and
    Settings\dlhawkin\Desktop\AUVMission.xsd">
    <Profile>Text</Profile>
    <InsertPoint Xcoordinate="5000" Ycoordinate="5000" Zcoordinate="164" />
        <Waypoints     number="0"     Xcoordinate="5000"     Ycoordinate="5000"
      Zcoordinate="164" />
    <StarboardPropSpeed>3820</StarboardPropSpeed>
    <PortPropSpeed>3820</PortPropSpeed>
    <Thrusters>1</Thrusters>
    <Rudder>90</Rudder>
    <ChangeCourse>359.9</ChangeCourse>
    <PlanesAngle stern="90" bow="90" both="90" />
    <CommandedAltitude Zcoordinate="164" />
    <CommandedDepth Zcoordinate="164" />
    <PitchAngle>30</PitchAngle>
    <Theta>30</Theta>
    <Rotate>40</Rotate>
    <Lateral>0.82</Lateral>
    <DiveTracker Xcoordinate="5000" Ycoordinate="5000" Zcoordinate="164" />
    <AltitudeOrDepthControl>1</AltitudeOrDepthControl>
```

```
    <PerformGPSPopup>1</PerformGPSPopup>
    <DurationGPSPopup>1000</DurationGPSPopup>
    <GyroError>180</GyroError>
    <DepthCellError>100</DepthCellError>
    <Position Xcoordinate="5000" Ycoordinate="5000" Zcoordinate="164" />
    <Orientation phi="30" theta="30" psi="359.9" />
     <Posture  Xcoordinate="5000"  Ycoordinate="5000"  Zcoordinate="164"  phi="30"
       theta="30" psi="359.9" />
    <OceanCurrent xAxis="50" yAxis="50" zAxis="50" />
    <SeaState>9</SeaState>
    <WatchRadius>10000</WatchRadius>
    <WaypointTimeout>1000</WaypointTimeout>
    <StandOffDistance>100</StandOffDistance>
     <Hover  enabled="1"  Xcoordinate="5000"  Ycoordinate="5000"  Zcoordinate="164"
       />
          <TargetStation      rangeToTarget="10000"      bearingToTarget="359.9"
       commandedRange="10000" commandedHeading="359.9" psi="359.9" />
    <TargetPoint>1</TargetPoint>
    <EnterTube range="20" bearing="359.9" />
    <Wait>1000</Wait>
    <WaitUntil>1000</WaitUntil>
    <TimeStep>1000</TimeStep>
    <SingleStep>1</SingleStep>
    <Pause>1</Pause>
    <RealTime>1</RealTime>
    <Virtual>Stringa</Virtual>
    <LocationLab>1</LocationLab>
    <Tethered>1</Tethered>
    <VirtualHost>Stringa</VirtualHost>
    <Mission>String</Mission>
    <Telemetry>String</Telemetry>
    <NoScript>1</NoScript>
    <Keyboard>1</Keyboard>
    <Trace>1</Trace>
    <TraceOn>1</TraceOn>
    <LoopForever>1</LoopForever>
    <ControlConstantsFilename>String</ControlConstantsFilename>
    <Text>1</Text>
    <Exit>1</Exit>
  - <SonarCommands>
        <Sonar725Installed     bearing="90"     range="10000"     power="50"
       direction="TRUE" />
    </SonarCommands>
    <Sound>1</Sound>
    <EMail>1</EMail>
    <SlidingModeCourse>1</SlidingModeCourse>
    <ParallelPortTrace>1</ParallelPortTrace>
    <ExtractPoint Xcoordinate="5000" Ycoordinate="5000" Zcoordinate="164" />
  </AUVMission>
```

Below is a sample mission script file for the AUV Workbench.

```
      <?xml version="1.0" encoding="UTF-8" ?>
      - <!--    edited  with  XMLSPY  v5  U  (http://www.xmlspy.com)  by  Douglas  Horner
(Naval Postgraduate School) -->
      -<AUVMission                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation=
"C:/AUVWorkbench/bin/scripts/missionScripts/AUVMission.xsd">
```

```
- <Profile>
- <InsertPoint>
  <Xcoordinate>-50</Xcoordinate>
  <Ycoordinate>10</Ycoordinate>
  </InsertPoint>
- <Waypoints number="1">
  <Xcoordinate>10</Xcoordinate>
  <Ycoordinate>10</Ycoordinate>
  <StarboardPropSpeed>2.75</StarboardPropSpeed>
  <PortPropSpeed>2.75</PortPropSpeed>
  <AltitudeOrDepthControl>0</AltitudeOrDepthControl>
  <CommandedAltitude>1.25</CommandedAltitude>
  <CommandedDepth>1.00</CommandedDepth>
  <PerformGPSPopup>0</PerformGPSPopup>
  <DurationGPSPopup>25</DurationGPSPopup>
  <WatchRadius>8</WatchRadius>
  <WaypointTimeout>40</WaypointTimeout>
  </Waypoints>
- <Waypoints number="2">
  <Xcoordinate>10</Xcoordinate>
  <Ycoordinate>210</Ycoordinate>
  <StarboardPropSpeed>2.75</StarboardPropSpeed>
  <PortPropSpeed>2.75</PortPropSpeed>
  <AltitudeOrDepthControl>0</AltitudeOrDepthControl>
  <CommandedAltitude>1.25</CommandedAltitude>
  <CommandedDepth>1.00</CommandedDepth>
  <PerformGPSPopup>0</PerformGPSPopup>
  <DurationGPSPopup>25</DurationGPSPopup>
  <WatchRadius>8</WatchRadius>
  <WaypointTimeout>200</WaypointTimeout>
  </Waypoints>
- <Waypoints number="3">
  <Xcoordinate>25</Xcoordinate>
  <Ycoordinate>210</Ycoordinate>
  <StarboardPropSpeed>2.75</StarboardPropSpeed>
  <PortPropSpeed>2.75</PortPropSpeed>
  <AltitudeOrDepthControl>0</AltitudeOrDepthControl>
  <CommandedAltitude>1.25</CommandedAltitude>
  <CommandedDepth>1.00</CommandedDepth>
  <PerformGPSPopup>0</PerformGPSPopup>
  <DurationGPSPopup>25</DurationGPSPopup>
  <WatchRadius>2</WatchRadius>
  <WaypointTimeout>15</WaypointTimeout>
  </Waypoints>
- <Waypoints number="4">
  <Xcoordinate>25</Xcoordinate>
  <Ycoordinate>10</Ycoordinate>
  <StarboardPropSpeed>2.75</StarboardPropSpeed>
  <PortPropSpeed>2.75</PortPropSpeed>
  <AltitudeOrDepthControl>0</AltitudeOrDepthControl>
  <CommandedAltitude>1.25</CommandedAltitude>
  <CommandedDepth>1.00</CommandedDepth>
  <PerformGPSPopup>0</PerformGPSPopup>
  <DurationGPSPopup>25</DurationGPSPopup>
  <WatchRadius>2</WatchRadius>
  <WaypointTimeout>200</WaypointTimeout>
  </Waypoints>
- <Waypoints number="5">
  <Xcoordinate>40</Xcoordinate>
  <Ycoordinate>10</Ycoordinate>
  <StarboardPropSpeed>2.75</StarboardPropSpeed>
  <PortPropSpeed>2.75</PortPropSpeed>
  <AltitudeOrDepthControl>0</AltitudeOrDepthControl>
```

```
        <CommandedAltitude>1.25</CommandedAltitude>
        <CommandedDepth>1.00</CommandedDepth>
        <PerformGPSPopup>0</PerformGPSPopup>
        <DurationGPSPopup>25</DurationGPSPopup>
        <WatchRadius>2</WatchRadius>
        <WaypointTimeout>15</WaypointTimeout>
        </Waypoints>
      - <Waypoints number="6">
        <Xcoordinate>40</Xcoordinate>
        <Ycoordinate>210</Ycoordinate>
        <StarboardPropSpeed>2.75</StarboardPropSpeed>
        <PortPropSpeed>2.75</PortPropSpeed>
        <AltitudeOrDepthControl>0</AltitudeOrDepthControl>
        <CommandedAltitude>1.25</CommandedAltitude>
        <CommandedDepth>1.00</CommandedDepth>
        <PerformGPSPopup>0</PerformGPSPopup>
        <DurationGPSPopup>25</DurationGPSPopup>
        <WatchRadius>2</WatchRadius>
        <WaypointTimeout>200</WaypointTimeout>
        </Waypoints>
      - <Waypoints number="7">
        <Xcoordinate>41</Xcoordinate>
        <Ycoordinate>210</Ycoordinate>
        <StarboardPropSpeed>2.75</StarboardPropSpeed>
        <PortPropSpeed>2.75</PortPropSpeed>
        <AltitudeOrDepthControl>0</AltitudeOrDepthControl>
        <CommandedAltitude>1.25</CommandedAltitude>
        <CommandedDepth>1.00</CommandedDepth>
        <PerformGPSPopup>0</PerformGPSPopup>
        <DurationGPSPopup>25</DurationGPSPopup>
        <WatchRadius>2</WatchRadius>
        <WaypointTimeout>1</WaypointTimeout>
        </Waypoints>
        <ExtractPoint />
        </Profile>
        </AUVMission>
```

Based on the mission command language schema document the AUV Workbench input file format, below is the XSLT used to create an date file that can be used by the AUV Workbench to execute a mission.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\Documents and
Settings\dlhawkin\Desktop\AUVMission.xsd">
   <xsl:output media-type="text/html" method="html" indent="yes" doctype-public="-
//W3C//DTD HTML 4.01//EN" doctype-system="http://www.w3.org/TR/html4/strict.dtd"/>
   <xsl:template match="/">
   <!-- Number of Waypoints -->
        <xsl:value-of select="count (//Waypoints)"/>
       <xsl:for-each select="//Waypoints">
       <br></br>
   <!-- Waypoints Xcoordinate -->
       <xsl:value-of select="./Xcoordinate"/>
      <xsl:text>  </xsl:text>
   <!-- Waypoints Ycoordinate -->
       <xsl:value-of select="./Ycoordinate"/>
      <xsl:text>  </xsl:text>
   <!-- StarboardPropSpeed -->
```

```
      <xsl:value-of select="./StarboardPropSpeed"/>
      <xsl:text>  </xsl:text>
   <!-- PortPropSpeed -->
      <xsl:value-of select="./PortPropSpeed"/>
      <xsl:text>  </xsl:text>
   <!-- Altitude or DepthControl -->
      <xsl:value-of select="./AltitudeOrDepthControl"/>
      <xsl:text>  </xsl:text>
   <!-- Commanded Altitude -->
      <xsl:value-of select="./CommandedAltitude"/>
      <xsl:text>  </xsl:text>
   <!-- Commanded Depth -->
      <xsl:value-of select="./CommandedDepth"/>
      <xsl:text>  </xsl:text>
   <!-- Perform GPS Popup -->
      <xsl:value-of select="./PerformGPSPopup"/>
      <xsl:text> </xsl:text>
   <!-- Duration of GPS Popup -->
      <xsl:value-of select="./DurationGPSPopup"/>
      <xsl:text>  </xsl:text>
   <!-- Watch Radius -->
      <xsl:value-of select="./WatchRadius"/>
      <xsl:text> </xsl:text>
   <!-- Waypoint Timeout -->
      <xsl:value-of select="./WaypointTimeout"/>
      </xsl:for-each>
       </xsl:template>
</xsl:stylesheet>
```

The mission script file is transformed by the XSLT to create the AUV Workbench input file below. The input file is in the exact format used by the AUV Workbench. By modifying the XSLT, not the AUV software, the user can format the data virtually in any form desired.

```
7
10 10 2.75 2.75 0 1.25 1.00 0 25 8 40
10 210 2.75 2.75 0 1.25 1.00 0 25 8 200
25 210 2.75 2.75 0 1.25 1.00 0 25 2 15
25 10 2.75 2.75 0 1.25 1.00 0 25 2 200
40 10 2.75 2.75 0 1.25 1.00 0 25 2 15
40 210 2.75 2.75 0 1.25 1.00 0 25 2 200
41 210 2.75 2.75 0 1.25 1.00 0 25 2 1
```

D. CHAPTER SUMMARY

Currently, the AUV Workbench is not compatible with all AUVs. Using the XSL and this mission command language, it is possible to generate virtually any type of data file the user desires. It follows that development of this language can be the key to the next level of commanding all AUVs.

THIS PAGE INTENTIONALLY LEFT BLANK

VII. THE BIGGER PICTURE – INTEGRATION OF XML AND GCCS/MEDAL

A. INTRODUCTION

Currently, the Navy requires that AUVs under development be able to export certain data in United States Message Text Format (USMTF). This requirement is so that the data obtained in a mission can be uploaded into the GCCS/MEDAL system, for use by all authorized Navy personnel.

B. GLOBAL COMMAND AND CONTROL SYSTEM (GCCS)

1. Overview

The Global Command and Control System (GCCS) is an automated system designed to support planning and become the single C4I system to support the warfighter. GCCS is the midterm solution to a C^4I for the Warrior (C^4IFTW) concept, which is committed to providing a joint system providing total battlespace information to the warrior. GCCS is a common operating environment (COE) that eliminates the need for stovepipe command and control systems. GCCS allows for the migration of existing systems into a COE connected across the SIPRNET and allows for the integration of C2 systems into an interoperable system. (After GCCS – Global Command and Control System – United States Nuclear Forces)

The first priority of GCCS is to become a globally connected, interoperable, fully integrated C4 system. Its common operational picture correlates and fuses data from multiple sources to provide the information needed to react decisively. GCCS enables joint force commanders to synchronize actions of multiple forces and has the flexibility to be used in operations from actual combat to humanitarian assistance. (After What is the Joint Command & Control System (GCCS-J))

2. Components

GCCS is made up of database servers, applications servers and clients. Connectivity is provided through the Defense Information System Network (DISN), and Secret connectivity is provided over the SIPRNET. (After What is the Joint Command &Control System (GCCS-J)) GCCS infrastructure includes a client server environment operating on an IEEE LAN, a GCCS Executive Subsystem (GES) that allows the user to launch GCCS applications, an Information

Management Subsystem (IMS), a Reference File Manager, and a communications capability. (After GCCS- Global Command and Control System – United States Nuclear System) GCCS only works on Hewlett-Packard (HP) workstations.

C. GLOBAL COMMAND AND CONTROL SYSTEM - MARITIME

GCCS- Maritime (GCCS-M), previously known as the Joint Maritime Command Information System (JMCIS), is the Navy's primary fielded command and control system. GCCS-M affords operational commanders the capability to receive, retrieve, and display information in a Common Operating Picture. (After Global Command and Control System - Maritime) GCCS-M developed over a number of years of various C4I initiatives, and evolved into a system, which allows applications to be run on a "superset" of core software. The core includes capabilities such as track and relational database management, tactical display, and communications interfaces. (After Module 8 – Intelligence Automated Data Processing System)

D. GCCS-M / MEDAL

1. Overview

The Mine Warfare Environmental Decision Aids Library (MEDAL) is one component of the GCCS-M. Like GCCS, MEDAL only works on Hewlett Packard workstations. Incorporation of the MEDAL into GCCS-M has strengthened the relationship between the MCM commander and the Carrier Battle Group (CVBG)/Amphibious Ready Group (ARG). MEDAL has increased the MCM Commander's contribution to the Common Operational Picture, and provides a coordinated MIW tactical picture. Using MEDAL, operators can import asset positions, contact positions, and environmental information, such as bathymetry, sound speed, and temperature and current data, and view the processed picture on screen.

2. Components

MEDAL contains several databases, including mine countermeasures, environmental and mine threat databases. These databases can be used for mine warfare (MIW) planning management, and are common and available to the entire navy. MEDAL allows the user to import a chart of an operational area with lanes and Q-routes. Once an area is defined, the user can plot planned and actual tracks, as well as asset positions. Contacts can be plotted with imbedded information and images.

In addition to databases, MEDAL contains area, contact, and asset directories. The area directory provides information about Q-routes and areas that have been cleared of mines. The contact directory lists all known contacts. Each is given a contact reference number (CRN) and, if judged by an Explosive Ordnance Disposal (EOD) Officer to be a mine, is given a mine reference number (MRN). The directory also includes the confidence level of identification, the position, and identification (e.g. bottom mine). Finally, the asset directory lists available assets, their tasking, and historical data points of tracks that have been run.

3. Using MEDAL with AUVs

Data can be entered into MEDAL one of two ways. First, the data can be entered by hand, which is an acceptable method for entering one contact, and is point-by-point. However, for numerous points, this method is very laborious, and so data can be entered automatically through a network connection. For example, messages can be sent into MEDAL via file transfer protocol (FTP) if they are in the proper United States Message Text Format (USMTF). In the various incoming logs (ILOGS), USMTF format is checked automatically, and if the message passes, it is immediately processed and the system updates, making the information available to all users.

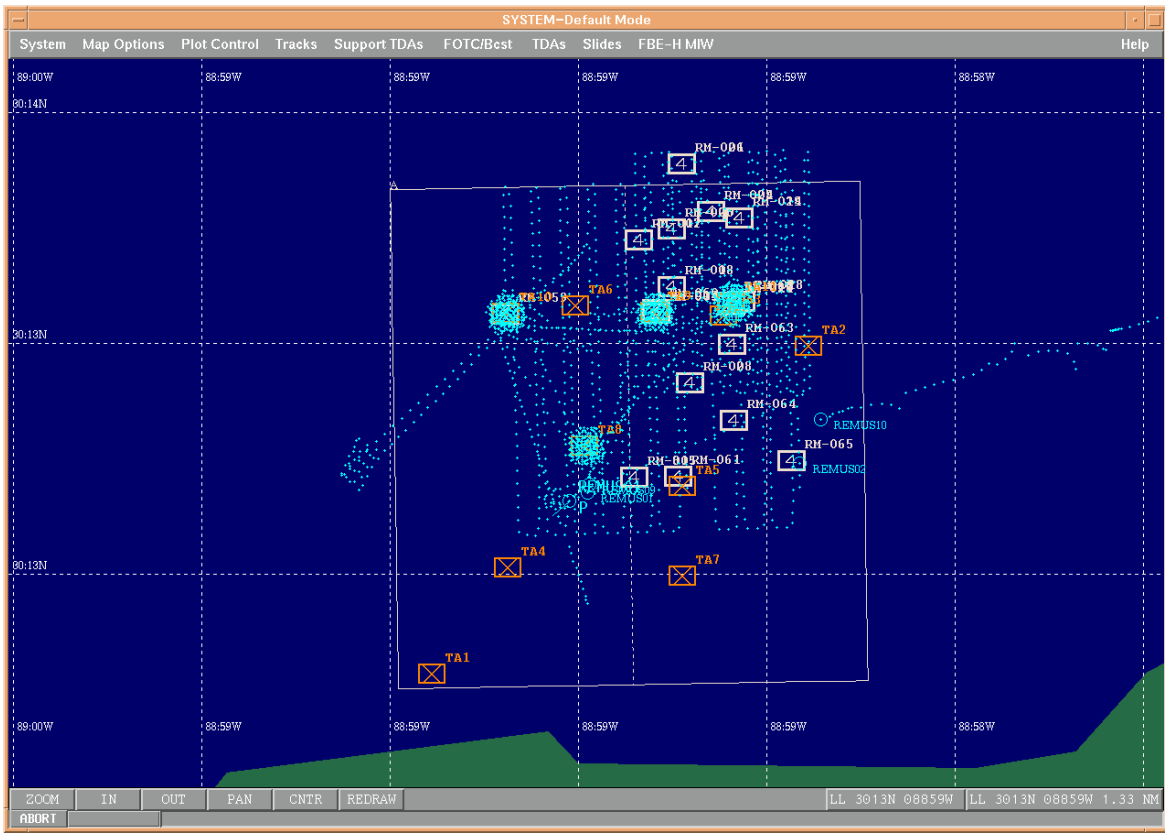

Figure 22. GCCS/MEDAL displaying Asset and Contact Positions. (From Weekley, 2003)

One of the problems with AUVs and MEDAL is that most AUVs store data in text files. Data stored on the hard drive of the AUV must first be converted into USMTF before it can be loaded into MEDAL. Some data collected by REMUS, such as asset and contact positions, and bathymetry information, can be exported from the AUV in USMTF format. This capability was a requirement of the vehicle when the Navy began the REMUS acquisition process. Most AUVs acquired by the Navy have this same requirement. The addition of these types of requirements equate to an increase in acquisition costs.

4. Solutions to AUV – MEDAL Incompatibilities

One way to change text data into USMTF format is to use the AUV Data Server (ADS). The ADS can manage the flow of data over a network, either by polling or by an operator. In doing so, the server increases the opportunities to manipulate and display data in new ways. One of these ways is by displaying the data in USMTF format. (After Weekley, 2003) Below is a screen shot of the ADS graphical user interface.

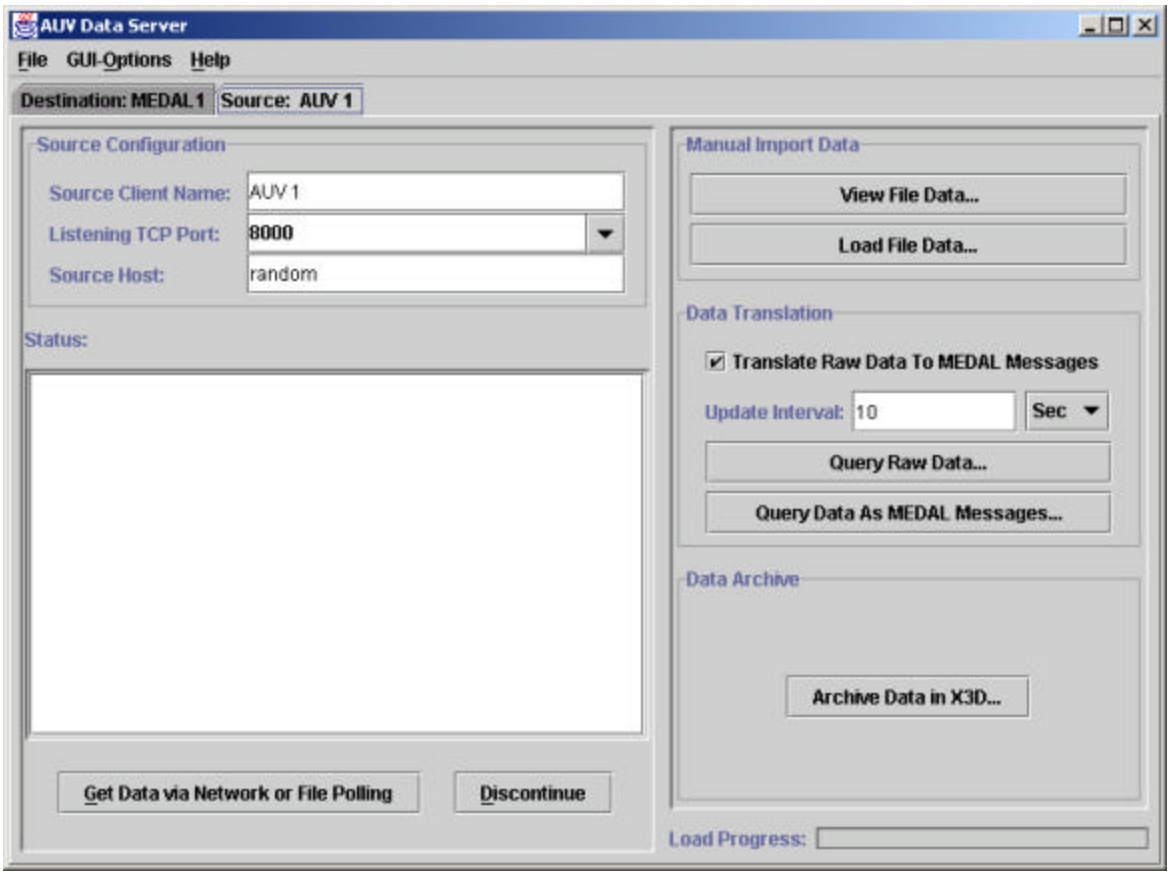

Figure 23. The ADS Graphical User Interface (From Weekley, 2003)

Another solution is to transform the stored text file into an XML document. Once in XML, a stylesheet can be used to transform the data into the desired format. At first glance, this solution appears to be less desirable than transforming the data directly into USMTF, because of the addition of the intermediate transformation into XML. However, the use of an intermediate XML document is beneficial for several reasons. First, XML documents must be well formed and validated against a schema. Existing software will check documents for structure and for validation against the governing schema. When a file does not match, the following type of error message occurs, and the error is highlighted.

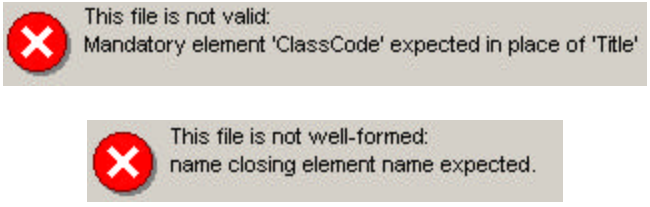

Figure 24. XML Spy Not Valid and Not Well-Formed Errors.

On the other hand, MEDAL offers only pass or fail when checking messages. If the message fails, there is no parser to automatically emphasize where the error occurred. In order to correct the message, the user must manually parse and correct the document, which can be very laborious and time consuming.

Another advantage of using XML over direct conversion to MEDAL is the ability to modernize the system. MEDAL is essentially running on technology that is 20 years old. To update the technology, one to five years of preparation, and Congressional permission are required, before the acquisition process even starts. A web year is said to be "the length of time it takes for the Internet technology to evolve as much as technology in another environment might evolve in a calendar year." (From Web Year) A web year is said to be about three months. Therefore, by the time MEDAL can be updated, the desired technology is already four to twenty years out of date. Also, USMTF files from early versions are no longer useful. Conversely, XML upgrades quite frequently. Also, all early version XML documents will validate under later versions. Therefore, while exporting data in USMTF format offers the luxury of a single transformation, exporting to XML offers much greater versatility, error-correction, and syntactic correctness.

E. SUMMARY

While the Navy currently requires that AUVs be able to export certain data in USMTF format for uploading into GCCS/MEDAL, this system is not always effective. Occasionally, the MEDAL message is not correctly formatted, and MEDAL will not accept the message. Also, MEDAL is not easily upgraded, and archiving information is expensive and impractical. XML is a viable solution to this problem. By exporting data gathered on a mission in XML format, the document can easily be transformed into USMTF format using XSLT, but can also be easily transformed into any necessary language, and can also be archived for later use.

THIS PAGE INTENTIONALLY LEFT BLANK

VIII. FUTURE CONCEPTS

A. UNDERWATER COMMUNICATIONS

Land-based digital communications are traditionally accomplished using radio or light-based transmissions. However, underwater communications pose a problem. (After Schrope, 2003) Long distance communications, such as those used for submarine contacts, can be accomplished on a low-frequency carrier, but the system is expensive, and the link is one-way. Additionally, these modems offer a data rates around 100 bits per second (bps). An alternative to radio and light wave communications is the acoustic channel. (After Schweber, 2001). The basic idea of underwater acoustics is to convert bits of information into tones, which are then converted back to digital data at the receiver. Because of the transmission problems introduced by underwater transmissions, the bits are sent as multiple tones to ensure the arrival of at least some. (After Schrope, 2003) Even as the redundancy in transmissions increases the chance that the message will arrive and be interpreted correctly, the redundancy makes the transfer of data even slower.

While the acoustic channel can be used over moderate distances of three to seven kilometers, increasing the data rate of the above long range communications by a factor of 24, the rate is still quite slow compared to land-based communications. In addition, land-based communications are almost 186,000 times faster than the speed of sound in water. (After Schrope 2003) Most AUVs collect some type of data, such as bathymetry, bottom type, contact positions, and asset positions. Ideally, this information can be transmitted to the command ship during the mission, without removing the AUV from its task. Though these files may be large, they typically contain data in the same format. Although large files, like images and sound, cannot be avoided, files containing similar types of data can be compressed using a technique called XML serialization.

B. XML SERIALIZATION

"Serialization is the process of converting an object into a form that can be readily transported." (From Introducing XML Serialization) This process is especially necessary for use with XML documents, because even a short XML document can quickly exceed the Maximum Transmission Unit (MTU), 1500 bytes for Ethernet. XML Serialization compacts XML

documents by replacing elements and attributes with specified tokens. Once serialized, the data can be passed, and then deserialized at the receiver. (After Serin, 2003)

Data compression typically minimizes the amount of data necessary to represent some information. Often referred to as coding, the objective of data compression is to represent source messages with corresponding code. XML uses markups to identify and describe data. While the markup structure is not economic, as stated in Chapter IV: terseness is of minimal importance, the characteristics of XML are fundamental for compression. Efficient representation of symbols leads to a decrease of the space needed to store it, and if the data is self-describing, data can be identified based on type and semantics. One system that has been developed for the compression of XML data is to use lossless compression techniques for the markup structure and both lossless and lossy techniques for the data itself. Lossless and lossy techniques refer to the techniques reversibility. Lossless compression means that decoded data are identical to original data; otherwise the compression technique is lossy.

C. FORWARD ERROR CORRECTION (FEC)

Forward Error Correction (FEC) is a "method of data encoding which gives the receiver the ability to correct data received in error up to a preset bound." According to thesis work performed in 1995, FEC can reduce the number of required retransmissions by 3 to 15 percent. The use of FEC is beneficial because acoustic shallow-water data transmissions are unreliable and an autonomous entity will often experience problems when passing a message to its intended reveicer. FEC is a beneficial solution to this 'retry until you die' syndrome, because it is easily implemented, the most basic implementation requiring the use of a simple Hamming code. (See Reimers, 1995 for more detail) As with the implementation of a XML-based mission control language, one goal of FEC is standardization of the underwater acoustic data communications community. (After Reimers, 1995)

D. USING SERIALIZATION TO IMPROVE UNDERWATER COMMUNICATIONS

Even though the speed of sound is almost five times faster in water than in air, the data transfer rate underwater does not compare to the data rate of land-based communications, which essentially travel at the speed of light. (After Schrope, 2003) Therefore, the ability to greatly decrease the size of the file to be transferred would be make for an improvement in the speed of

underwater communications. By programming the AUVs to export data in XML format, the data could be serialized and then transferred at a much better rate.

E. SEMANTIC WEB AND APPLICATIONS

The Semantic Web is the "representation of data on the World Wide Web. It is a collaborative effort led by [World Wide Web Consortium (W3C)] with participation from a large number of researchers and industrial partners."(From W3C Semantic Web) Rather than being a separate Web, the Semantic Web will be an extension of the existing World Wide Web. Through the conceptual Semantic Web, information is given "well-defined meaning, better enabling computers and people to work in cooperation." (From Scientific American, 2001) One important technology needed for the development of the Semantic Web that is already in place is XML. XML allows the user to "add arbitrary structure to their documents but says nothing about what the structures mean." (From Scientific American, 2001)

The idea of the Semantic Web is to make data on the web available to programs and machines, much the way it is available to people. The Semantic Web applications can also be applied to data coming to and from AUVs. Currently, data from an AUV script file is readable by humans. However, machines cannot process those same files. By exporting that data in a machine readable, validatable format, such as XML, the data can be archived, and reused months or years later, by machines or programs, without a human in the loop. In addition, a validatable file can be checked for completeness and correctness, vice just completeness.

F. SECURITY APPLICATIONS

In order to address the security issues created by XML, the W3C has created a recommendation for security and authentication technologies called XML Digital Signatures.(After Deitel, 2001) "Digital signatures provide integrity, signature assurance, and non-repudiatability over Web data." (From Digital Signature Activity Statement, June 2003) These features are especially important to documents that contain such information as contracts, price lists, and manifests, and can be applied to use in transmission of data to and from AUVs. Much of the data used in conjunction with the vehicles is sensitive material, and security of this information is critical.

Cryptology, a branch of applied mathematics concerned with transforming messages in to unintelligible forms and back again, is used to create and verify digital signatures. Digital

signatures can take the form of a public or secret key system. With a secret key, both the sender and receiver must have the key to verify the information. However, with a public key, a sender can sign a piece of information and anyone possessing the public key can authenticate that information. (From Digital Signature Activity Statement, June 2003)

As mentioned above, a goal for AUVs is to be able to perform wireless communications between an AUV on a mission and a master AUV, or a data collecting station. In these types of situations, security becomes a major issue. Security of AUV missions may become important in tactical situations. Security is needed both through the water, and over the Internet when orders are transmitted. Acoustic communications are fundamentally insecure and low bandwidth. However, encryption and authentication, in the form of digital signatures and key distribution, are already specified for XML. Thus, an XML based mission command language has a readily available, no-cost security capability. As always, any use of security in the underwater environment will be case dependant and require careful implementation.

G. SUMMARY

Many opportunities for future work arise with the use of XML for AUVs. As always, underwater communications require compressed files, in order to overcome problems due to low bandwidth, and also redundancy and forward error correction to overcome the amount of loss and interference experienced during underwater communications. The Semantic Web is the new frontier for web applications and the use of XML as a mission command language for AUVs begins to make the inclusion of AUVs and the data collected by AUVs on the Semantic Web possible. Finally, the use of acoustic communications creates many security issues that can be cheaply and easily addressed using XML.

IX. CONCLUSIONS AND RECOMMENDATIONS

According to Tim Berners-Lee's keynote address at the WWW 2003 Symposium in Budapest, Hungary May 2003, there are three stages to new users adopting XML. The first stage is what the heck is this stuff, and why is it useful for anything? Next, the user decides he will use XML, but he doesn't have to understand or like it. Finally, the user picks up his laptop and tells everyone, yelling, "Look at this! The whole world is here on my laptop!" The purpose of this thesis was to bring the reader at least to stage two. Numerous conclusions and recommendations for future worth follow

A. CONCLUSIONS: AUVS AND XML

Currently, each type of AUV, e.g. REMUS, BPAUV, RMS, and LMRS, has a distinct acquisition program and area of operation. While each vehicle performs similar operations, they are unable to communicate without a human in the loop. One solution, introduced in this thesis, is to create a common mission and data formatting language, using XML. Once AUVs become more prevalent in the Navy's work, commands are likely to be using multiple vehicles, for multiple depth ranges. Without a common mission and data formatting language, multiple vehicles mean multiple operators, or one operator, well versed in several computer-programming languages. Not only is XML simple to learn, relative to other computer languages, it also offers several other advantages. First, an XML document must be well formed, meaning that is must have syntactic correctness. While other languages will not operate properly without proper syntax, none offers the immediate highlighting of the syntax error that any parser will give in an XML document. Second, XML documents must validate against a schema. In the schema, the operator can specify ranges for values or can require that the user enter one of a number of specified options. Using a tool like XML Spy, the user can validate their mission document against the schema, and highlight any mistakes. For example, if the vehicle references depth to the water's surface and operates with depth as a positive number, the schema can be designed to only allow positive values for depth. If a user tries to enter a negative number for depth, the schema will not validate, and the error will be highlighted.

Not only must XML be well formed, but the tags must also match those specified in the schema, both spelling and case. Once again, an error with the tag will prevent the XML

document from validating. This feature of XML makes the need for redundant commands unnecessary. Currently, the vehicle languages offer several synonyms of each command in the hopes of avoiding a misspelled command. The best-case scenario of a misspelled command is that the vehicle does not complete the assigned mission. The worst-case scenario of a misspelled word is that the vehicle shuts down, sinks and is never seen again.

B. THE BIGGER PICTURE

Current AUV acquisition programs require that the vehicle be able to output certain information, such as bathymetry, bottom type, and contact and asset position, in USMTF format. The stored data can be exported directly from the vehicle's hard drive, over a network, into GCCS-M/MEDAL. In addition, the ADS can take a mission data file, strip out the desired data, and convert it into USMTF format. However, neither of these methods allow for quick parsing and error correction. When a message is uploaded into MEDAL, it is automatically checked for proper format and then given a pass or fail. In order to correct a failed message, it is sometimes necessary to delete and recreate huge chunks of data, which can be time-consuming and tedious. Using XML, a message will not validate, unless well formed, and in the format required by the schema. A simple XSLT style sheet can then be used to transform the document into USMTF format, for uploading into MEDAL. Again, a simple parser immediately finds any errors.

Another reason that exporting in XML is an improvement over exporting in USMTF and directly into the MEDAL system is that XML keeps relative pace with the advancement of technology. While XML updates occur frequently, an update to the MEDAL system requires Congressional approval, and anywhere from one to five years of red tape. By comparison, a web year is approximately three months, so by the time the MEDAL update is approved, the technology is already four to 20 years out of date.

C. RECOMMENDATIONS FOR FUTURE WORKS AND CONCEPTS

The Semantic Web is an extension of the current web, in which information is "given well-defined meaning, better enabling computers and people to work in cooperation." (From Berners-Lee, Hendler, Lassila) XML is an important technology already in place for the Semantic Web, because it allows users to add arbitrary structure to their documents without saying what the structures mean. (After Berners-Lee) In the future, commands will likely be dealing with fleets of AUVs. With the implementation of XML as a common mission and data

formatting language, the Semantic Web may be used as a means of communication between AUVs, and a store of data, in addition to the MEDAL system.

Another potential for XML in the underwater realm is the use of XML Serialization. XML Serialization and the recently developed Cross Format Schema Protocol (XFSP) make it possible to compress a very large file into a much smaller one, via tokenization of element tags and attributes. This compression is highly desirable in the operation of AUVs, because of the need for wireless underwater communications. The density of water inhibits the travel of radio and light waves, while sound travels quite well. However, the data rates remain poor in comparison to land-based communications. The compression of data files would be beneficial to the speed of data transfer to and from AUVs.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A ABBREVIATIONS

API	**Application Programming Interface**
ARG	**Amphibious Ready Group**
ARIES	**Acoustic Radio Information Exchange Server**
AUVs	**Autonomous Underwater Vehicles**
BPAV	**Battlespace Preparation Autonomous Underwater Vehicle**
bps	**bits per second**
C^2	**Command and Control**
C^4I	**Command & Control, Communications, Computers and Intelligence**
C^4IFTW	**C^4I for the Warrior**
CLZ	**Coastal Landing Zone**
COE	**Common Operating Environment**
COI	**Community of Interest**
CRN	**Contact Reference Number**
CSS	**Cascading Style Sheets**
CSS2	**Cascading Style Sheets, level 2**
CVBG	**Carrier Battle Group**
DISN	**Defense Information System Network**
DTD	**Document Type Declaration**
EOD	**Explosive Ordnance Disposal**
FEC	**Forward Error Correction**
FTP	**File Transfer Protocol**
GCCS	**Global Command and Control System**
GCCS-M	**GCCS- Maritime**
GES	**GCCS Executive Subsystem**
HTML	**Hypertext Markup Language**
ILOGS	**Incoming Logs**
IMS	**Information Management Subsystem**
JMCIS	**Joint Maritime Command Information System**
LMRS	**Long-Term Mine Reconnaissance System**
MCM	**Mine Countermeasures**
MEDAL	**Mine Warfare Environmental Decision Aids Library**
MIW	**Mine Warfare**
MRN	**Mine Reference Number**
MTU	**Maximum Transmission Unit**
NPS	**Naval Postgraduate School**
RDBMS	**Relational Database Management System**
REMUS	**Remote Environment Monitoring Units**
RMS	**Remote Minehunting System**
SGML	**Standard Generalized Markup Language**
SME	**Subject Matter Expert**
UAV	**Unmanned Aerial Vehicle**
USMTF	**United States Message Text Format**
UUV	**Unmanned Undersea Vehicle**

VRML	**Virtual Reality Modeling Language**
W3C	**World Wide Web Consortium**
WWW	**World Wide Web**
X3D	**Extensible 3D Graphics**
XFSP	**Cross Format Schema Protocol**
Xj3D	**Extensible Java 3D Graphics**
XML	**Extensible Markup Language**
XSL	**Extensible Stylesheet Language**
XSL-FO	**XSL Formatting Objects**
XSLT	**Extensible Stylesheet Language for Transformations**

APPENDIX B AUV MISSION COMMAND AND TELEMETRY LANGUAGE DEFINITIONS: XML SCHEMA

```
<?xml version="1.0" encoding="UTF-8" ?>
- <!-- edited with XMLSPY v5 rel. 2 U (http://www.xmlspy.com) by Barbara Van Leuvan (Naval Postgraduate School) -->
  - <!--
      This schema describes the AUV mission scripting. Refer to the mission.script.Help file for a description of the commands. Similar command were consolidated and the assumptions made regarding data validation.
                  This schema is modification of a schema created by Daniel Kucik. Darrin L. Hawkins and Barbara Van Leuvan will use it to aid in their thesis work to create a common Mission and data formatting language for Autonomous Underwater Vehicles (AUVs). -->
- <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified">
  - <xs:annotation>
      <xs:documentation>Start of defining data types</xs:documentation>
    </xs:annotation>
  - <xs:simpleType name="absoluteHeadingType">
    - <xs:annotation>
        <xs:documentation>Defines valid absolute heading values</xs:documentation>
      </xs:annotation>
    - <xs:restriction base="xs:decimal">
        <xs:minInclusive value="0" />
        <xs:maxInclusive value="359.9" />
      </xs:restriction>
    </xs:simpleType>
  - <xs:simpleType name="depthType">
    - <xs:annotation>
        <xs:documentation>Defines valid commanded depth values</xs:documentation>
      </xs:annotation>
    - <xs:restriction base="xs:decimal">
        <xs:minInclusive value="0" />
        <xs:maxInclusive value="164" />
      </xs:restriction>
    </xs:simpleType>
  - <xs:simpleType name="depthErrorType">
    - <xs:annotation>
        <xs:documentation>Defines valid values for indicating depth cell errors</xs:documentation>
      </xs:annotation>
    - <xs:restriction base="xs:decimal">
        <xs:minInclusive value="-100" />
        <xs:maxInclusive value="100" />
      </xs:restriction>
    </xs:simpleType>
```

```
- <xs:annotation>
    <xs:documentation>Can dive tracker parameters be defined by XYZ
      coordinate attribute group or are the values
      different</xs:documentation>
  </xs:annotation>
- <xs:simpleType name="diveTrackerDepthType">
  - <xs:annotation>
      <xs:documentation>Defines valid dive tracker depth
        values</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="0" />
      <xs:maxInclusive value="100" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="diveTrackerXPositionType">
  - <xs:annotation>
      <xs:documentation>Defines valid values for dive tracker x-
        position</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-5000" />
      <xs:maxInclusive value="5000" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="diveTrackerYPositionType">
  - <xs:annotation>
      <xs:documentation>Defines valid values for dive tracker y-
        position</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-5000" />
      <xs:maxInclusive value="5000" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="fileNameType">
  - <xs:annotation>
      <xs:documentation>Defines valid file names</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:string">
      <xs:minLength value="1" />
      <xs:maxLength value="100" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="gyroErrorType">
  - <xs:annotation>
      <xs:documentation>Defines valid values for indicating gyro
        offset/error</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
```

```
      <xs:minInclusive value="-180" />
      <xs:maxInclusive value="180" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="hostNameType">
  - <xs:annotation>
      <xs:documentation>Defines valid host
      names</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:string">
      <xs:minLength value="7" />
      <xs:maxLength value="100" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="lateralTranslationRateType">
  - <xs:annotation>
      <xs:documentation>Defines valid lateral translation
      rates</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-0.82" />
      <xs:maxInclusive value="0.82" />
    </xs:restriction>
  </xs:simpleType>
- <xs:annotation>
    <xs:documentation>Defines latitude and longitude
    types</xs:documentation>
  </xs:annotation>
- <xs:simpleType name="latitudeType">
  - <xs:annotation>
      <xs:documentation>Defines valid latitude values (+/-
      ddmm.mmmmm)</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-9000" />
      <xs:maxInclusive value="9000" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="longitudeType">
  - <xs:annotation>
      <xs:documentation>Defines valid longitude values (+/-
      dddmm.mmmmm)</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-18000" />
      <xs:maxInclusive value="18000" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="pitchAngleType">
```

```
- <xs:annotation>
            <xs:documentation>Defines valid pitch
      angles</xs:documentation>
  </xs:annotation>
- <xs:restriction base="xs:decimal">
    <xs:minInclusive value="-30" />
    <xs:maxInclusive value="30" />
  </xs:restriction>
</xs:simpleType>
- <xs:simpleType name="planeAngleType">
- <xs:annotation>
      <xs:documentation>Defines valid angles for the
      planes</xs:documentation>
  </xs:annotation>
- <xs:restriction base="xs:decimal">
    <xs:minInclusive value="-90" />
    <xs:maxInclusive value="90" />
  </xs:restriction>
</xs:simpleType>
- <xs:simpleType name="propSpeedType">
- <xs:annotation>
        <xs:documentation>Defines valid prop speed
      values</xs:documentation>
  </xs:annotation>
- <xs:restriction base="xs:decimal">
    <xs:minInclusive value="-3820" />
    <xs:maxInclusive value="3820" />
  </xs:restriction>
</xs:simpleType>
- <xs:simpleType name="rangeType">
- <xs:annotation>
            <xs:documentation>Defines valid range
      values</xs:documentation>
  </xs:annotation>
- <xs:restriction base="xs:decimal">
    <xs:minInclusive value="0" />
    <xs:maxInclusive value="10000" />
  </xs:restriction>
</xs:simpleType>
- <xs:simpleType name="relativeHeadingType">
- <xs:annotation>
      <xs:documentation>Defines valid relative heading
      values</xs:documentation>
  </xs:annotation>
- <xs:restriction base="xs:decimal">
    <xs:minInclusive value="-359.9" />
    <xs:maxInclusive value="359.9" />
  </xs:restriction>
</xs:simpleType>
```

```
- <xs:simpleType name="relativeXPositionType">
  - <xs:annotation>
      <xs:documentation>Defines valid values for relative x-coordinates</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-5000" />
      <xs:maxInclusive value="5000" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="relativeYPositionType">
  - <xs:annotation>
      <xs:documentation>Defines valid values for relative y-coordinates</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-5000" />
      <xs:maxInclusive value="5000" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="rollAngleType">
  - <xs:annotation>
      <xs:documentation>Defines valid roll angles</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-30" />
      <xs:maxInclusive value="30" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="rotationRateType">
  - <xs:annotation>
      <xs:documentation>Defines valid rates of rotation</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-40" />
      <xs:maxInclusive value="40" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="rudderAngleType">
  - <xs:annotation>
      <xs:documentation>Defines valid rudder angle values</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-90" />
      <xs:maxInclusive value="90" />
    </xs:restriction>
  </xs:simpleType>
```

```
- <xs:simpleType name="seaStateType">
  - <xs:annotation>
      <xs:documentation>Defines valid sea state values</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:integer">
      <xs:minInclusive value="0" />
      <xs:maxInclusive value="9" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="st725BearingType">
  - <xs:annotation>
      <xs:documentation>Defines valid relative heading values for the ST-725 sonar</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-90" />
      <xs:maxInclusive value="90" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="st725PowerType">
  - <xs:annotation>
      <xs:documentation>Defines valid power values for the ST-725 sonar</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="0" />
      <xs:maxInclusive value="50" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="st725RangeType">
  - <xs:annotation>
      <xs:documentation>Defines valid ST-725 Sonar range values</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="0" />
      <xs:maxInclusive value="10000" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="st1000BearingType">
  - <xs:annotation>
      <xs:documentation>Defines valid relative heading values for the ST-1000 sonar</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-90" />
      <xs:maxInclusive value="90" />
    </xs:restriction>
```

```
  </xs:simpleType>
- <xs:simpleType name="st1000SweepWidthType">
  - <xs:annotation>
      <xs:documentation>Defines valid scan widths for the ST-1000 sonar</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-90" />
      <xs:maxInclusive value="90" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="st725SweepWidthType">
  - <xs:annotation>
      <xs:documentation>Defines valid scan widths for the ST-725 sonar</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-90" />
      <xs:maxInclusive value="90" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="standoffDistanceType">
  - <xs:annotation>
      <xs:documentation>Defines valid values for stand off distances</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="0" />
      <xs:maxInclusive value="100" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="timeClockType">
  - <xs:annotation>
      <xs:documentation>Defines valid AUV clock values</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="0" />
      <xs:maxInclusive value="1000" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="timeSecondsType">
  - <xs:annotation>
      <xs:documentation>Defines valid time values in seconds</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="0" />
      <xs:maxInclusive value="1000" />
```

```
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="tubeHeadingType">
  - <xs:annotation>
      <xs:documentation>Defines valid tube heading (orientation)
       values</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="0" />
      <xs:maxInclusive value="359.9" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="tubeRangeType">
  - <xs:annotation>
          <xs:documentation>Defines valid tube range
       values</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="0" />
      <xs:maxInclusive value="20" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="trueRelativeEnumType">
  - <xs:annotation>
       <xs:documentation>Defines valid direction indicators
       (true/relative headings)</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:string">
      <xs:enumeration value="TRUE" />
      <xs:enumeration value="true" />
      <xs:enumeration value="True" />
      <xs:enumeration value="RELATIVE" />
      <xs:enumeration value="relative" />
      <xs:enumeration value="Relative" />
    </xs:restriction>
  </xs:simpleType>
- <xs:simpleType name="waterCurrentRateType">
  - <xs:annotation>
      <xs:documentation>Defines valid values for indicating water
       current</xs:documentation>
    </xs:annotation>
  - <xs:restriction base="xs:decimal">
      <xs:minInclusive value="-50" />
      <xs:maxInclusive value="50" />
    </xs:restriction>
  </xs:simpleType>
- <!--
=============================================================
  ========================
  -->
```

```
- <xs:annotation>
    <xs:documentation>Start of defining complex types, which will
     make-up the elements contained in root element
     AUVMission.</xs:documentation>
  </xs:annotation>
- <xs:complexType name="DepthPositionType">
  - <xs:attribute name="Zcoordinate" type="depthType">
    - <xs:annotation>
        <xs:documentation>Z coordinate (depth) of the commanded
         waypoint</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
- <xs:attributeGroup name="EnterTubeAttributes">
  - <xs:annotation>
      <xs:documentation>Defines the parameter structure for the
       EnterTube Command</xs:documentation>
    </xs:annotation>
  - <xs:attribute name="range" type="tubeRangeType">
    - <xs:annotation>
        <xs:documentation>How far forward to travel to be fully
         inside tube</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  - <xs:attribute name="bearing" type="tubeHeadingType">
    - <xs:annotation>
        <xs:documentation>Tube orientation in
         degrees</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:attributeGroup>
- <xs:attributeGroup name="PlanesAttributes">
  - <xs:annotation>
      <xs:documentation>Defines the parameter structure for setting
       the plane angles</xs:documentation>
    </xs:annotation>
  - <xs:attribute name="stern" type="planeAngleType">
    - <xs:annotation>
        <xs:documentation>Stern plane angle</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  - <xs:attribute name="bow" type="planeAngleType">
    - <xs:annotation>
        <xs:documentation>Bow plane angle</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  - <xs:attribute name="both" type="planeAngleType" use="optional">
    - <xs:annotation>
        <xs:documentation>Set both the stern and bow plane
         angles equal to the given angle</xs:documentation>
```

```
      </xs:annotation>
    </xs:attribute>
  </xs:attributeGroup>
- <xs:attributeGroup name="XYZCoordinates">
  - <xs:annotation>
      <xs:documentation>Defines the parameter structure for setting the AUV's posture (position and orientation)</xs:documentation>
    </xs:annotation>
  - <xs:attribute name="Xcoordinate" type="relativeXPositionType">
    - <xs:annotation>
        <xs:documentation>Current X position</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  - <xs:attribute name="Ycoordinate" type="relativeYPositionType">
    - <xs:annotation>
        <xs:documentation>Current Y position</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  - <xs:attribute name="Zcoordinate" type="depthType" use="optional">
    - <xs:annotation>
        <xs:documentation>Current Z position</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:attributeGroup>
- <xs:attributeGroup name="Angles">
  - <xs:attribute name="phi" type="rollAngleType">
    - <xs:annotation>
        <xs:documentation>Current roll angle</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  - <xs:attribute name="theta" type="pitchAngleType">
    - <xs:annotation>
        <xs:documentation>Current pitch angle</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  - <xs:attribute name="psi" type="absoluteHeadingType">
    - <xs:annotation>
        <xs:documentation>Current heading</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:attributeGroup>
- <xs:complexType name="Sonar725Type">
  - <xs:annotation>
      <xs:documentation>Defines the parameter structure used for the ST-725 Sonar</xs:documentation>
    </xs:annotation>
    <xs:attribute name="bearing" type="st725BearingType" />
  - <xs:attribute name="range" type="st725RangeType">
```

```
    - <xs:annotation>
        <xs:documentation>Sonar range</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  - <xs:attribute name="power" type="st725PowerType">
    - <xs:annotation>
        <xs:documentation>Sonar power</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  - <xs:attribute name="direction" type="trueRelativeEnumType">
    - <xs:annotation>
        <xs:documentation>Direction type for bearing (true or
          relative)</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
- <xs:complexType name="Sonar1000Type">
  - <xs:annotation>
      <xs:documentation>Defines the parameter structure used for
        the ST-1000 Sonar</xs:documentation>
    </xs:annotation>
    <xs:attribute name="bearing" type="st1000BearingType" />
  - <xs:attribute name="direction" type="trueRelativeEnumType">
    - <xs:annotation>
        <xs:documentation>Direction type for bearing (true or
          relative)</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
- <xs:annotation>
    <xs:documentation>Define basic structure for all commands used to
      set speed</xs:documentation>
  </xs:annotation>
- <xs:complexType name="SpeedType">
  - <xs:annotation>
      <xs:documentation>Defines the parameter structure for setting
        prop speeds</xs:documentation>
    </xs:annotation>
  - <xs:attribute name="StarboardPropSpeed" type="propSpeedType">
    - <xs:annotation>
                    <xs:documentation>Starboard prop
        speed</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  - <xs:attribute name="PortPropSpeed" type="propSpeedType">
    - <xs:annotation>
        <xs:documentation>Port prop speed</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  - <xs:attribute name="both" type="propSpeedType" use="optional">
```

```
    - <xs:annotation>
        <xs:documentation>Set both port and starboard props to
          given value</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
- <xs:attributeGroup name="StationAttributes">
  - <xs:annotation>
      <xs:documentation>Defines the parameter structure for the
        stationkeeping commands</xs:documentation>
    </xs:annotation>
  -   <xs:attribute name="rangeToTarget" type="rangeType"
      use="required">
    - <xs:annotation>
        <xs:documentation>Range to sonar
          target</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  -  <xs:attribute name="bearingToTarget" type="absoluteHeadingType"
      use="required">
    - <xs:annotation>
        <xs:documentation>Bearing to sonar
          target</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  -   <xs:attribute name="commandedRange" type="rangeType"
      use="optional">
    - <xs:annotation>
        <xs:documentation>Commanded range</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  -  <xs:attribute name="commandedHeading"
      type="absoluteHeadingType" use="optional">
    - <xs:annotation>
        <xs:documentation>Commanded
          heading</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  -   <xs:attribute name="psi" type="absoluteHeadingType"
      use="optional">
    - <xs:annotation>
        <xs:documentation>Commanded AUV
          heading</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:attributeGroup>
- <xs:attributeGroup name="WaterCurrentAttributes">
  - <xs:annotation>
      <xs:documentation>Defines the parameter structure used to
        describe water currents</xs:documentation>
```

```
    </xs:annotation>
  -     <xs:attribute      name="xAxis"      type="waterCurrentRateType"
      use="required">
    - <xs:annotation>
        <xs:documentation>Water current in the north-south
        direction</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  -     <xs:attribute      name="yAxis"      type="waterCurrentRateType"
      use="required">
    - <xs:annotation>
        <xs:documentation>Water current in the east-west
        direction</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  -     <xs:attribute      name="zAxis"      type="waterCurrentRateType"
      use="optional">
    - <xs:annotation>
        <xs:documentation>Water current in the up-down
        direction</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:attributeGroup>
- <!--
=============================================================
  ========================
  -->
- <xs:element name="AUVMission">
  - <xs:annotation>
      <xs:documentation>AUV Mission Script</xs:documentation>
    </xs:annotation>
  - <xs:complexType>
    - <xs:sequence maxOccurs="unbounded">
        <xs:element name="Profile" />
      - <xs:element name="InsertPoint">
        - <xs:annotation>
            <xs:documentation>Where the vehicle enters the
            water.</xs:documentation>
          </xs:annotation>
        - <xs:complexType>
            <xs:attributeGroup ref="XYZCoordinates" />
          </xs:complexType>
        </xs:element>
      - <xs:element name="Waypoints" minOccurs="0">
        - <xs:annotation>
            <xs:documentation>Drive to the given
            waypoint</xs:documentation>
          </xs:annotation>
        - <xs:complexType>
          - <xs:attribute name="number" type="xs:integer">
```

```
      - <xs:annotation>
            <xs:documentation>ID number of the
            commanded
            waypoint</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attributeGroup ref="XYZCoordinates" />
    </xs:complexType>
  </xs:element>
- <xs:element name="StarboardPropSpeed"
  type="propSpeedType">
  - <xs:annotation>
      <xs:documentation>Starboard prop
      speed</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="PortPropSpeed"
  type="propSpeedType">
  - <xs:annotation>
      <xs:documentation>Port prop
      speed</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="Thrusters" type="xs:boolean"
  minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Enable vertical and lateral
      thruster control</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="Rudder" type="rudderAngleType"
  minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Set the rudder angle (thrusters
      off)</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="ChangeCourse"
  type="relativeHeadingType" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Adjust heading by the given
      number of degrees (positive for starboard and
      negative for port)</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="PlanesAngle" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Set the angle of the planes
      (thrusters off)</xs:documentation>
    </xs:annotation>
```

```
- <xs:complexType>
    <xs:attributeGroup ref="PlanesAttributes" />
  </xs:complexType>
</xs:element>
- <xs:element name="CommandedAltitude"
  type="DepthPositionType" minOccurs="0">
- <xs:annotation>
    <xs:documentation>Set a new ordered
    altitude.</xs:documentation>
  </xs:annotation>
</xs:element>
- <xs:element name="CommandedDepth"
  type="DepthPositionType" minOccurs="0">
- <xs:annotation>
    <xs:documentation>Set a new ordered
    depth.</xs:documentation>
  </xs:annotation>
</xs:element>
- <xs:element name="PitchAngle" type="pitchAngleType"
  minOccurs="0">
- <xs:annotation>
    <xs:documentation>Set a new ordered pitch
    angle.</xs:documentation>
  </xs:annotation>
</xs:element>
- <xs:element name="Theta" type="pitchAngleType"
  minOccurs="0">
- <xs:annotation>
    <xs:documentation>Set a new ordered pitch
    angle.</xs:documentation>
  </xs:annotation>
</xs:element>
- <xs:element name="Rotate" type="rotationRateType"
  minOccurs="0">
- <xs:annotation>
    <xs:documentation>Open loop lateral thruster
    rotation control at the given rate
    (degrees/sec).</xs:documentation>
  </xs:annotation>
</xs:element>
- <xs:element name="Lateral"
  type="lateralTranslationRateType" minOccurs="0">
- <xs:annotation>
    <xs:documentation>Open loop lateral thruster
    translation control in given ft/sec. (positive is
    to starboard, max is ~0.82 ft/sec) Thruster
    orders are constrained to +/- 24.0 volts = 3820
    rpm no-load interestingly some yaw occurs in
    open-loop control.</xs:documentation>
  </xs:annotation>
```

```
  </xs:element>
- <xs:element name="DiveTracker" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Position of DiveTracker transducer 1.</xs:documentation>
    </xs:annotation>
  - <xs:complexType>
      <xs:attributeGroup ref="XYZCoordinates" />
    </xs:complexType>
  </xs:element>
- <xs:element name="AltitudeOrDepthControl" type="xs:boolean" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Indicated whether Altitude or Depth Control is on or off</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="PerformGPSPopup" type="xs:boolean" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>GPS fix complete, resume previously ordered depth.</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="DurationGPSPopup" type="timeSecondsType" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Duration, in seconds, to proceed to shallow depth, take GPS fix, restore ordered depth when done.</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="GyroError" type="gyroErrorType" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Degrees of error measured for gyro-compassGYRO + ERROR = TRUE</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="DepthCellError" type="depthErrorType" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Feet of bias error measured for depth cell.DEPTH CELL Z + BIAS = TRUE Z</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="Position" minOccurs="0">
  - <xs:annotation>
```

```
        <xs:documentation>Reset vehicle's dead reckoning
          position.</xs:documentation>
      </xs:annotation>
    - <xs:complexType>
        <xs:attributeGroup ref="XYZCoordinates" />
      </xs:complexType>
    </xs:element>
  - <xs:element name="Orientation" minOccurs="0">
    - <xs:annotation>
                <xs:documentation>Reset          vehicle's
          orientation</xs:documentation>
      </xs:annotation>
    - <xs:complexType>
        <xs:attributeGroup ref="Angles" />
      </xs:complexType>
    </xs:element>
  - <xs:element name="Posture" minOccurs="0">
    - <xs:annotation>
          <xs:documentation>Reset    vehicle's    posture
          (position and orientation)</xs:documentation>
      </xs:annotation>
    - <xs:complexType>
        <xs:attributeGroup ref="XYZCoordinates" />
        <xs:attributeGroup ref="Angles" />
      </xs:complexType>
    </xs:element>
  - <xs:element name="OceanCurrent" minOccurs="0">
    - <xs:annotation>
                <xs:documentation>Ocean          current
          rate</xs:documentation>
      </xs:annotation>
    - <xs:complexType>
        <xs:attributeGroup ref="WaterCurrentAttributes" />
      </xs:complexType>
    </xs:element>
  -    <xs:element     name="SeaState"     type="seaStateType"
      minOccurs="0">
    - <xs:annotation>
        <xs:documentation>Estimate of surface sea state
          [0-9]</xs:documentation>
      </xs:annotation>
    </xs:element>
  -    <xs:element     name="WatchRadius"     type="rangeType"
      minOccurs="0">
    - <xs:annotation>
        <xs:documentation>Circular area that if the ARIES
          gets inside that distance it has achieved it's
          goal      of      navigating      to      the
          waypoint</xs:documentation>
      </xs:annotation>
```

```
  </xs:element>
- <xs:element name="WaypointTimeout"
    type="timeSecondsType" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Time in seconds until the
        ARIES will discontinue navigating toward that
        waypoint.</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="StandOffDistance"
    type="standoffDistanceType" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Change standoff distance for
        WAYPOINT and HOVER
        control</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="Hover" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Hover at present (or provided)
        position and depth</xs:documentation>
    </xs:annotation>
  - <xs:complexType>
    - <xs:annotation>
        <xs:documentation>Defines the parameter
          structure for the hovering
          commands</xs:documentation>
      </xs:annotation>
    - <xs:attribute name="enabled" type="xs:boolean">
      - <xs:annotation>
          <xs:documentation>Hover mode is on or
            off</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attributeGroup ref="XYZCoordinates" />
    </xs:complexType>
  </xs:element>
- <xs:element name="TargetStation" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Hover relative to sonar target
        at the given range and bearing with AUV
        pointing at the target. Stationkeeping will use
        full target tracking sonar
        mode.</xs:documentation>
    </xs:annotation>
  - <xs:complexType>
      <xs:attributeGroup ref="StationAttributes" />
    </xs:complexType>
  </xs:element>
```

```
- <xs:element name="TargetPoint" type="xs:boolean"
  minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Commanded Psi during
      stationkeeping will point directly at target
      center</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="EnterTube" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Experimental control mode.
      This tells execution level that nose has entered
      the tube, drive the rest of the way in using
      dead reckon for forward motion and sonars
      (pointing to opposite sides) to maintain tube
      side wall standoff.</xs:documentation>
    </xs:annotation>
  - <xs:complexType>
      <xs:attributeGroup ref="EnterTubeAttributes" />
    </xs:complexType>
  </xs:element>
- <xs:element name="Wait" type="timeSecondsType"
  minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Wait for #a seconds prior to
      reading from the script file
      again</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="WaitUntil" type="timeClockType"
  minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Wait (or run) until robot clock
      reaches time #a (letting the AUV execute its
      current orders) prior to reading from the script
      file again. If #a is earlier than current time,
      reset the clock. If in TACTICAL mode, command
      is ignored.</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="TimeStep" type="timeSecondsType"
  minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Change default execution level
      time step interval from default of 0.1 sec to the
      provided number of
      seconds</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="SingleStep" type="xs:boolean"
  minOccurs="0">
```

```
- <xs:annotation>
    <xs:documentation>Loop for another timestep prior
      to reading script again. Only useful in execution
      keyboard mode.</xs:documentation>
  </xs:annotation>
</xs:element>
- <xs:element name="Pause" type="xs:boolean"
  minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Stop execution until (enter) is
        pressed</xs:documentation>
    </xs:annotation>
</xs:element>
- <xs:element name="RealTime" type="xs:boolean"
  minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Run execution level code in
        real-time(busy wait at the end of each timestep
        if time remains)</xs:documentation>
    </xs:annotation>
</xs:element>
- <xs:element name="Virtual" type="hostNameType"
  minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Tells the execution level to
        open a socket to the virtual world which is
        already running and waiting on #a.
        VIRTUALHOST is a command line
        switch.</xs:documentation>
    </xs:annotation>
</xs:element>
- <xs:element name="LocationLab" type="xs:boolean"
  minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Vehicle is operating in lab
        using virtual world. This is the default
        mode.</xs:documentation>
    </xs:annotation>
</xs:element>
- <xs:element name="Tethered" type="xs:boolean"
  minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Command line switch only,
        used for in-water runs</xs:documentation>
    </xs:annotation>
</xs:element>
- <xs:element name="VirtualHost" type="hostNameType"
  minOccurs="0">
  - <xs:annotation>
```

```
        <xs:documentation>Tells the execution level to
          open a socket to the virtual world which is
          already running and waiting on #a.
          VIRTUALHOST is a command line
          switch.</xs:documentation>
      </xs:annotation>
    </xs:element>
-   <xs:element name="Mission" type="fileNameType"
    minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Replace temporary file
      “mission.script” with the given and start the
      new mission. Reads tactical commands for
      execution level from the given
      file.</xs:documentation>
    </xs:annotation>
  </xs:element>
-   <xs:element name="Telemetry" type="fileNameType"
    minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Playback prerecorded
      telemetry data from the given file. Consider
      using with NOSCRIPT if no script file is
      present</xs:documentation>
    </xs:annotation>
  </xs:element>
-   <xs:element name="NoScript" type="xs:boolean"
    minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Ignore script command file.
      Selectively used in combination with
      TELEMETRY data file
      playback.</xs:documentation>
    </xs:annotation>
  </xs:element>
-   <xs:element name="Keyboard" type="xs:boolean"
    minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Read script commands from
      keyboard</xs:documentation>
    </xs:annotation>
  </xs:element>
-   <xs:element name="Trace" type="xs:boolean"
    minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Enable verbose print
      statements in execution
      level</xs:documentation>
    </xs:annotation>
  </xs:element>
```

```
- <xs:element name="LoopForever" type="xs:boolean"
  minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Repeat current mission
      indefinitely. Each repetition is called a
      “replication”. Do not generate plots after each
      replication</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="ControlConstantsFilename"
  type="fileNameType" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Read revised control
      coefficients from the given
      file.</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="Text" type="xs:boolean" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Turn text display in command
      window on or off</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="Exit" type="xs:boolean" minOccurs="0">
  - <xs:annotation>
      <xs:documentation>Do not execute any more
      commands in this script, but repeat the mission
      again if LOOP-FOREVER is
      set.</xs:documentation>
    </xs:annotation>
  </xs:element>
- <xs:element name="SonarCommands" minOccurs="0">
  - <xs:complexType>
    - <xs:choice>
      - <xs:element name="Sonar725Installed"
        type="Sonar725Type" minOccurs="0">
        - <xs:annotation>
            <xs:documentation>Use the installed
            sonar</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:element name="Sonar1000Installed"
        type="Sonar1000Type" minOccurs="0">
        - <xs:annotation>
            <xs:documentation>Use the installed
            sonar</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:choice>
    </xs:complexType>
```

```
        </xs:element>
      - <xs:element name="Sound" type="xs:boolean"
          minOccurs="0">
        - <xs:annotation>
            <xs:documentation>Enable text-to-speech audio
              output, on or off</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:element name="EMail" type="xs:boolean"
          minOccurs="0">
        - <xs:annotation>
            <xs:documentation>Ask user for e-mail address at
              start of mission. E-mail report once mission is
              complete.</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:element name="SlidingModeCourse" type="xs:boolean"
          minOccurs="0">
        - <xs:annotation>
            <xs:documentation>Sliding mode course control
              algorithm</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:element name="ParallelPortTrace" type="xs:boolean"
          minOccurs="0">
        - <xs:annotation>
            <xs:documentation>Enable trace statements for
              parallel port
              communications</xs:documentation>
          </xs:annotation>
        </xs:element>
      - <xs:element name="ExtractPoint">
        - <xs:annotation>
            <xs:documentation>Where the vehicle is taken out
              of the water.</xs:documentation>
          </xs:annotation>
        - <xs:complexType>
            <xs:attributeGroup ref="XYZCoordinates" />
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D SOFTWARE AVAILABILITY

1. INTRODUCTION

This appendix describes the availability and installation for most of the software packages used to generate this thesis. This section provides instructions for downloading and installing the software. To create the Common Mission and Data Formatting Language schema, one may use XMLSpy. The NPS AUV Workbench is discussed in this thesis and an example mission script file in taken from the NPS AUV Workbench to show interoperability with existing software. Thus, the installation of the NPS AUV Workbench is included.

2. XML-BASED COMMON MISSION AND DATA FORMATTING LANGUAGE

To view the XML-based Common Mission and Data Formatting Language, one must have a XML editor. This section describes the software availability and an overview of the installation instructions. The common XML editor of choice is XMLSpy because it's the industry standard XML Development Environment for building software applications based on XML technologies. XMLSpy (version 5) is available at http://www.altova.com/download. Download the executable file and follow the installation instructions.

3. NPS AUV WORKBENCH

The NPS AUV Workbench is designed to provide a virtual environment for planning for and analyzing AUV operations. The idea is to provide mission scripts, obstacle fields, sonar algorithms and an ocean environment and use the Xj3D Browser together with X3D internet graphics for representing the virtual environment. (Note: This is the initial version of the AUV Workbench. It isn't well documented or fully functionally, use at your own risk!)

The best way to install the software is to use the Install Anywhere package available through the Naval Postgraduate School. This will automatically create a .exe and load the APIs necessary to run the scenarios. If you only have the .zip file do the following:

Unzip the AUVWorkbench.zip into a C:/AUVWorkbench directory.

Make sure that you have J2SDK1.4.1 loaded and working

Install the JDOM API, check examples to make sure it is working.

At a command window change the directory to C:/AUVWorkbench and type AMVW_1.0. (This should start the AUVWorkbench GUI program.

In the Xj3DBrowser click on the open button and open the file AUVMissionScenario.wrl. This gives a predesigned AUV Mission.

A note on the architecture. You can create any type on mission profile and obstacles on the workbench. Missions are created by importing the various component XML files into a master scenario that file is then converted into a X3D file for rendering in the Xj3D browser. In other words, go to the Route Planner, open or create an XML mission script and press the import button to bring the file into the master scenario. Do the same for the obstacle field. Once you have selected and imported the component XML files, open the X3D file via the top Xj3D Browser and choose the file GeneralAUVMissionScenario.wrl. This will be the file that runs the specific mission.

Send comments questions to dphorner@nps.navy.mil

APPENDIX E – MISSION SCRIPT HELP FILE

```
//--------------------------------------------------------------------------//
        mission.script.HELP                     10 January 2002
        Mission script syntax for NPS AUV execution level and tactical
        control, in water and in the NPS AUV Underwater Virtual World.
        http://web.nps.navy.mil/~brutzman/auv/execution/mission.script.HELP
        Don Brutzman                    brutzman@nps.navy.mil
//--------------------------------------------------------------------------//
This file describes how to change and create NPS AUV mission script files.
            Example mission.script files and the 'execution' program are in the
            ~/execution subdirectory.
Script commands are received by the AUV execution level (execution.c) from
            the tactical level during a mission, the operator at the keyboard, or
            read from the "mission.script" file.  Both tactical and execution can
            carry out mission scripts.
To run a new mission, copy a different existing mission file over file
            'mission.script' or edit the mission.script file for a new mission.
            Example:
            unix> cd execution
            unix> cp  mission.script.siggraph  mission.script
            unix> execution  virtual cadet.cs.nps.navy.mil
  or, more simply,
            unix> execution  virtual cadet  mission  mission.script.siggraph
Many of the following commands will also work when invoked from the command
            line upon execution.  Detailed command line guidance is also available
            interactively using the online NPS AUV process launcher form at
            http://blackand.stl.nps.navy.mil/~auv/launcher/launcher.cgi
Numerous script keywords (and synonyms) are currently recognized.  We have been
            generous in the use of synonyms in order to reduce the possibility of
            catastrophic spelling errors.  This approach might be further extended
            to include synonyms in other languages (French, Portuguese etc.)
            Hint hint!
Sections in this syntax help file:
            - Helm commands:  open-loop and closed-loop control
            - Navigation commands
            - Mission timing commands
            - Mission setup and configuration commands
            - Sonar commands
            - Miscellaneous commands
```

```
        |          |
---------+------------+-------------------------------------------------------
Keywords | Parameters |  Description
Synonyms | [optional] |  (all units are feet, degrees or seconds as appropriate)
---------+------------+-------------------------------------------------------
        |          |
// Helm commands:  open-loop and closed-loop control -------------------------//
RPM                 # [##]     Set ordered rpm values to # for both propellers

SPEED               # [##]     [ or independently set left & right rpm values

PROPS               # [##]      to # and ## respectively]
PROPELLORS     # [##]     maximum propellor speed is +- 700 rpm => 2 ft/sec
    /* constrain thruster orders  +/- 24.0 volts == 3820 rpm no-load        */
THRUSTERS-ON            Enable vertical and lateral thruster control
THRUSTERS                       Thruster orders are constrained to
THRUSTERON                      +/- 24.0 volts == 3820 rpm no-load
                                Default turn-on voltage 0.0
THRUSTERSON

NOTHRUSTER              Disable vertical and lateral thruster control
NOTHRUSTERS
THRUSTERS-OFF
THRUSTERSOFF

RUDDER                  #       Force rudder to remain at # degrees, thrusters-off.
                Value is for after rudder, negative command turns left.

DEADSTICKRUDDER     [#]         Force rudder to remain at 0 [or #] degrees,
                thrusters-off.
COURSE                  #       Set new ordered course (commanded yaw angle)
HEADING                 #
YAW                 #
TURN                #           Change ordered course by # degrees
CHANGE-COURSE       #                       (positive # to starboard, negative # to port)
PLANES              # [#]       Force planes to remain at # degrees, thrusters-off.
PLANE          # [#]   Value is for stern planes, negative command points down.
DEADSTICKPLANES # [#]   Note that negative stern planes results in reduction in
z (i.e. more shallow).  If two values are applied, order is          PLANES stern bow.
Thus, for example:
```

PLANES –10 is equivalent to
PLANES -10 10 # stern=-10, bow=10, go shallow

DEADSTICKPLANES [#] Force planes to remain at 0 [or #] degrees, thrusters-off.

DEPTH # Set new ordered depth (commanded z)

PITCH # Set new ordered pitch (commanded theta angle).
THETA # Only effective during HOVERCONTROL.

ROTATE # open loop lateral thruster rotation control
at # degrees/sec

NOROTATE disable open loop lateral thruster rotation control
ROTATEOFF
ROTATE-OFF

LATERAL # open loop lateral thruster translation control
in # ft/sec.
(positive is to starboard, max is ~0.82 ft/sec)
Thruster orders are constrained to
+/- 24.0 volts == 3820 rpm no-load
interestingly some yaw occurs in open-loop control.

NOLATERAL disable open loop lateral thruster translation control
LATERALOFF
LATERAL-OFF

VERTICAL needed!

//----------//
// Navigation commands ----------//

DIVETRACKER1 # ## ### Position of DiveTracker transducer 1
DIVETRACKER2 # ## ### Position of DiveTracker transducer 2
Still need to incorporate bearing to DiveTrackers.

GPS Proceed to shallow depth, take Global Positioning
GPSFIX System (GPS) fix, restore ordered depth when done.

GPS-FIX Control (thrusters, propellers/planes, combined)
is not modified. Maximum fix time is 30 seconds,
at which time execution returns to previously
ordered depth.

GPS-COMPLETE GPS fix complete, resume previously ordered depth.
GPS-FIX-COMPLETE

GYRO-ERROR # Degrees of error measured for gyrocompass.
GYROERROR # [GYRO + ERROR = TRUE]

DEPTH-CELL-BIAS # Feet of bias error measured for depth cell.
DEPTHCELLBIAS # [DEPTH CELL Z + BIAS = TRUE Z]
DEPTH-CELL-ERROR #
DEPTHCELLERROR #

POSITION # ## [###] reset vehicle dead reckon position to (x, y) or
LOCATION # ## [###] (x, y, z) = (#, ##, ###) at current clock time
FIX # ## [###] This is a navigational position fix. Receipt of a
POSITION/LOCATION/FIX command resets the execution
level dead-reckon position. Note that depth value z
will likely be reset by depth cell if operational.
During virtual world operation, hydrodynamics model
is rezeroed.

ORIENTATION # ## ### reset vehicle orientation to
ROTATION # ## ### (phi, theta, psi) = (#, ##, ###)
During virtual world operation, hydrodynamics model
is rezeroed.

POSTURE #a #b #c #d #e #f
reset vehicle dead reckon posture to
(x, y, z, phi, theta, psi) = (#a, #b, #c, #d, #e, #f)

OCEANCURRENT #x #y [#z] Ocean current rate along North-axis, East-axis and
OCEAN-CURRENT #x #y [#z] [optional] Depth-axis (feet/sec)
(this is cartesian version of parametric set and drift)
During virtual world operation, hydrodynamics model
is rezeroed.

SEASTATE # Estimate of surface sea state, rounds to integer [0..9]
SEA-STATE # This value is also passed to dynamics level.

WAYPOINT #X #Y [#Z] [#rpm]
WAYPOINT-ON #X #Y [#Z] [#rpm]
Point towards waypoint with coordinates (#X, #Y) (depth #Z optional) (speed #rpm optional). You can leave waypoint control by ordering course, rudder, sliding-mode, rotate or lateral thruster control.

If speed is < 200 RPM, port & starboard RPMs are increased to 400 RPM to ensure waypoint can be achieved.

If in TACTICAL mode, execution reports STABLE when waypoint is achieved.

STANDOFF # Change standoff distance for WAYPOINT and HOVER
STAND-OFF # control. Default value is 2.5 feet for NPS AUV,
STANDOFFDISTANCE # 50.0 feet for SSN. Default values are automatically
STANDOFF-DISTANCE # read from control.constants.input.hulltype files.
STAND-OFF-DISTANCE #

HOVER Hover at present position and ordered depth using thrusters and propellers.

HOVER without parameters is the preferred method of slowing since backing down with negative propellers may result in large sternway and severe depth excursions.

HOVER [#X #Y] [#Z] Hover using thrusters and propellers for lateral and longitudinal positioning at specified position. Default Z value is previously ordered DEPTH.

HOVER [#X #Y] [#Z] [#orientation] [#standoff-distance]

Uses WAYPOINT control until within #standoff-distance of HOVER point (#X, #Y, #Z), then switches to HOVER control with [optional] final #orientation

Full speed (700 RPM) port & starboard is used if
AUV distance to WAYPOINT is > #standoff-distance + 10',
then slows to 200 RPM until within #standoff-distance,
then HOVER control.

If in TACTICAL mode, execution reports STABLE when done.

HOVEROFF — Turn off HOVER mode
HOVER-OFF
HOVER_OFF

TARGETSTATION #R #B [#Psi]
TARGET-STATION #R #B [#Psi]
Hover relative to a sonar target at range = #R and
target bearing #B from the AUV. Commanded AUV
heading is #Psi (default is point at target).
Stationkeeping will use full target tracking
sonar mode

TARGETSTATION #R1 #B1 #R2 #B2 [#Psi]
TARGET-STATION #R1 #B1 #R2 #B2 [#Psi]
Hover relative to sonar target. Target currently
at range = #R1, bearing #B1 from AUV. Commanded
range = #R2, commanded bearing = #B2, commanded
heading = #Psi (default is point at target).
Stationkeeping will use full target tracking
sonar mode

EDGESTATION #R #B [#Psi]
EDGE-STATION #R #B [#Psi]
Hover relative to a sonar target at range = #R and
target bearing #B from the AUV. Commanded AUV
heading is #Psi (default is point at target).
Stationkeeping will use full target tracking
sonar mode

EDGESTATION #R1 #B1 #R2 #B2 [#Psi]
EDGE-STATION #R1 #B1 #R2 #B2 [#Psi]
Hover relative to sonar target. Target currently
at range = #R1, bearing #B1 from AUV. Commanded

range = #R2, commanded bearing = #B2, commanded heading = #Psi (default is point at target).
Stationkeeping will use target edge tracking sonar mode

TARGET-OFF TARGETOFF NO-TARGET NOTARGET	Turn off stationkeeping control mode
TARGET-POINT TARGETPOINT	Commanded #Psi during stationkeeping will point directly at target center
NO-TARGET-POINT NOTARGETPOINT TARGET-POINT-OFF TARGETPOINTOFF	Commanded #Psi during stationkeeping can be manually controlled using HEADING commands
ENTERTUBE # ## ENTER-TUBE # ##	Experimental control mode. This tells execution level that nose has entered the tube, drive the rest of the way in using dead reckon for forward motion and sonars (pointing to opposite sides) to maintain tube side wall standoff. Parameters: # How far forward to travel to be fully inside tube ## Tube orientation in degrees

//----------//
// Mission timing commands ----------//

WAIT RUN	#	Wait (or run) for # seconds (letting the robot execute) prior to reading from the script file again

If in TACTICAL mode, execution ignores WAIT commands.

NEXTORDERTIME WAITUNTIL PAUSEUNTIL TIME	#	Wait (or run) until robot clock reaches time # (letting the robot execute its current orders) prior to reading from the script file again

** If value is earlier than current time, reset the clock.

** If in TACTICAL mode, execution ignores these commands.

TIMESTEP	#	change default execution level time step interval
TIME-STEP	#	from default of 0.1 sec to # sec

STEP	loop for another timestep prior to reading script again.
SINGLE-STEP	Only useful in execution keyboard mode.

PAUSE	temporarily stop execution until <enter> is pressed

REALTIME	run execution level code in real-time
REAL-TIME	(busy wait at the end of each timestep if time remains)

NOREALTIME	run execution level code as quickly as possible
NO-REALTIME	
NONREALTIME	
NOWAIT	
NO-WAIT	
NOPAUSE	
NO-PAUSE	

//---//
// Mission setup and configuration commands ----------------------------------//

HELP	Provide a list of available keywords
?	(as specified in this HELP file).
/?	
-?	

//	comments follow on this line which are not executed
/*	note comments will still be spoken if AUDIO-ON
#	pound sign also indicates a comment if in first column

// Three startup modes:	[LOCATIONLAB] | TETHERED | UNTETHERED

LOCATIONLAB	Vehicle is operating in lab using virtual world.
LOCATION-LAB	This is default mode.

TETHER	command line switch only, used for in-water runs
TETHERED	set DISPLAYSCREEN=TRUE and LOCACTIONLAB=FALSE

UNTETHER	command line switch only, used for in-water runs
UNTETHERED	set DISPLAYSCREEN=FALSE and LOCACTIONLAB=FALSE
NOTETHER	
NO-TETHER	

VIRTUAL hostname	tells execution level to open socket to virtual world
VIRTUALHOST hostname	which is already running and waiting on 'hostname'
REMOTE hostname	VIRTUALHOST is a command line switch. Example:
REMOTEHOST hostname	unix> execution virtualhost cadet.stl.nps.navy.mil
DYNAMICS hostname	

TACTICAL hostname	tells execution level to open socket to tactical level
TACTICALHOST hostname	which is already running and waiting on 'hostname'
STRATEGIC hostname	TACTICAL/STRATEGIC is a command line switch. Example:
STRATEGICHOST hostname	unix> execution tacticalhost cadet.stl.nps.navy.mil

MISSION filename	Replace temporary file 'mission.script' with 'filename'
SCRIPT filename	and start the new mission. Reads tactical commands for
FILE filename	execution level from 'filename.'
	Option: on SGI you can abbreviate, e.g. you can type
	mission siggraph
	instead of
	mission mission.script.siggraph

TELEMETRY filename	Playback prerecorded telemetry data from filename.
	Consider using with NOSCRIPT if no script file present.
	examples:
	execution telemetry mission.output.1_second
	execution telemetry mission.output.telemetry
	dynamics should be run with selection
	E dEad_reckon_test_with_execution_level
	or command line
	dynamics telemetry

NOSCRIPT	Ignore script command file. Selectively used
	in combination with TELEMETRY data file playback.

KEYBOARD	read script commands from keyboard

KEYBOARD-ON

KEYBOARD-OFF read script commands from mission.script file
NO-KEYBOARD

TRACE enable verbose print statements in execution level
TRACE-ON

TRACEOFF disable verbose print statements in execution level
TRACE-OFF
NOTRACE
NO-TRACE

LOOPFOREVER repeat current mission when done, indefinitely.
LOOP each repetition is called a 'replication.'
LOOP-FOREVER do not generate plots after each replication.

LOOPONCE do not LOOPFOREVER, stop when end of script is reached
LOOP-ONCE

LOOPFILEBACKUP back up output files during each loop replication
LOOP-FILE-BACKUP to permit inspection while new files are written
the backup files are in execution directory:
output.telemetry.previous & output.1_second.previous

CONSTANTS-FILE filename read revised control coefficients from "filename"
CONSTANTS filename i.e. control.constants.input.auv, ..ssn, ..suboff
and overwrite default file control.constants.input

ENTERCONTROLCONSTANTS use keyboard to enter revised control coefficients
ENTER-CONTROL-CONSTANTS
ENTER-CONSTANTS
ENTERCONSTANTS

SHOWCONTROLCONSTANTS display control coefficients
SHOW-CONTROL-CONSTANTS
SHOW-CONSTANTS
SHOWCONSTANTS

BENCH-TEST Simplified initial command-line parameter for quick

BENCHTEST switch setting during Russ's control and prop testing.
BENCH

NOTEXT Eliminate text display in command window
NO-TEXT (useful for verbose/long runs in virtual world)

TEXT Turn text display in command window back on
TEXT-ON

QUIT do not execute any more commands in this script, but
STOP repeat the mission again if LOOP-FOREVER is set
DONE
EXIT
REPEAT
RESTART
COMPLETE
<eof> marker

KILL same as QUIT but also shuts down socket to virtual world
SHUTDOWN 'dynamics' process.

//---//
// Sonar commands --//

SONAR725 #b [#r #p #d] Set the bearing (#b), range (#r), and power (#p) of the
SONAR-725 #b [#r #p #d] ST725 sonar. In virtual world, bearing is necessary for
SONAR_725 #b [#r #p #d] sonar model. In water, this stores data in the execution
ST725 #b [#r #p #d] level state vector for replay and examination. ST725 is
ST-725 #b [#r #p #d] electronically controlled by the tactical level laptop.
ST_725 #b [#r #p #d] Optional [#d] direction: TRUE or RELATIVE

SONAR1000 #b [#d] Manually control the 000 sonar bearing to #b degrees
SONAR-1000 #b [#d] relative to Phoenix heading. ST1000 is electronically
SONAR_1000 #b [#d] controlled by the execution level Gespac serial port.
ST1000 #b [#d]
ST-1000 #b [#d] Optional [#d] direction: TRUE or RELATIVE
ST_1000 #b [#d]

ST1000-SCAN-WIDTH # Total degrees for default sweep sonar scan, centered
ST1000SCANWIDTH # about bow

ST725-SCAN-WIDTH #
ST725SCANWIDTH #

SONARTRACE Enable verbose print statements in execution sonar code

SONARTRACEOFF Disable verbose print statements in execution sonar code

SONARINSTALLED Sonar interface(s) installed, use them
SONAR-INSTALLED
ST1000-INSTALLED
ST725-INSTALLED

NOSONARINSTALLED Sonar interface(s) not installed, don't use them
NO-SONAR-INSTALLED
NO-ST1000-INSTALLED
NO-ST725-INSTALLED

//----------//
// Miscellaneous commands ----------//

AUDIBLE enable text-to-speech audio output
AUDIO
AUDIO-ON
SOUND-ON
SOUNDON
SOUND

SILENT disable text-to-speech audio output
SILENCE
NOSOUND
SOUNDOFF
SOUND-OFF
AUDIOOFF
AUDIO-OFF
QUIET

SOUNDSERIAL tell virtual world to pause while playing back sound
SOUND-SERIAL (default)

SOUNDPARALLEL tell virtual world to play sounds as parallel processes

SOUND-PARALLEL (this may cause garbles if speeches play simultaneously)

EMAIL ask user for electronic mail address at mission start,
EMAIL-ON send user an electronic mail report at mission finish
E-MAIL
E-MAIL-ON
EMAILON

EMAILOFF disable electronic mail address query feature
EMAIL-OFF
E-MAILOFF
E-MAIL-OFF
NO-E-MAIL
NO-EMAIL
NO-E-MAIL
NOEMAIL

SLIDINGMODECOURSE Sliding mode course control algorithm (not yet working)
SLIDING-MODE-COURSE

SLIDINGMODEOFF Disable sliding mode course control algorithm (" " ")
SLIDING-MODE-OFF

PARALLELPORTTRACE enable trace statements for parallel port communications

WAYPOINTFOLLOW Deprecated, no longer needed, do not use.
WAYPOINTFOLLOWOFF

//----------//

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX F – PROPOSED AUV NAMESPACE

// Helm commands: open-loop and closed-loop control ----------//

PropellorSpeed		
	Definition:	Set ordered rpm values to # for both propellers [or independently set left & right rpm values to # and ## respectively] maximum propellor speed is +- 700 rpm => 2 ft/sec /* constrain thruster orders +/- 24.0 volts == 3820 rpm no-load
	Attributes:	starboard port both (optional)
	Modifications	Propellor is misspelled in schema
Thrusters		
	Definition	If value = 1: Enable vertical and lateral thruster control orders are constrained to +/- 24.0 volts == 3820 rpm no-load Default turn-on voltage 0.0 If value = 0: Disable vertical and lateral thruster control
Rudder		
	Definition:	Force rudder to remain at # degrees, thrusters-off. Value is for after rudder, negative command turns left. Force rudder to remain at 0 [or #] degrees, thrusters-off.
ChangeCourse		
	Definition	Change ordered course by # degrees (positive # to starboard, negative # to port)
	Modifications	No course tag exists with original ordered course
PlanesAngle		
	Definition	Force planes to remain at # degrees, thrusters-off. Value is for stern planes, negative command points down. Note that negative stern planes results in reduction in z (i.e. more shallow). If two values are applied, order is PLANES stern bow. Thus, for example: PLANES –10 is equivalent to PLANES -10 10 # stern=-10, bow=10, go shallow
	Modifications	Mission Script Help file allows for DEADSTICKPLANES mode in which the planes are forced to remain at 0 or specified degrees, with thrusters off.
	Attributes:	stern bow both (optional)
Depth		
	Definition	Set new ordered depth (commanded z)
	Attribute:	zposition
PitchAngle		
	Definition	Set new ordered pitch (commanded theta angle). Only effective during HOVERCONTROL.
Theta		
	Definition	Same as PitchAngle
	Modifications	Same as Pitch Angle
Rotate		
	Definition	open loop lateral thruster rotation control at # degrees/sec disable open loop lateral thruster rotation control
	Modifications	Needs an attribute to allow for disabled/enabled
Lateral		
	Definition	open loop lateral thruster translation control in # ft/sec. (positive is to starboard, max is ~0.82 ft/sec) Thruster orders are constrained to +/- 24.0 volts == 3820 rpm no-load interestingly some yaw occurs in open-loop control.
	Modifications	Attribute is needed to enable/disable open loop lateral thruster translation control
DiveTracker		
	Definition	Position of DiveTracker transducer
	Attributes	
GPSFixComplete		
	Definition	Proceed to shallow depth, take Global Positioning System (GPS) fix, restore ordered depth when done. Control (thrusters, propellers/planes, combined) is not modified. Maximum fix time is 30 seconds, at which time execution returns to previously ordered depth. GPS fix complete, resume previously ordered depth.

Attributes
Modifications

GyroError

Definition — Degrees of error measured for gyrocompass. [GYRO + ERROR = TRUE]
Attributes
Modifications

DepthCellError

Definition — Feet of bias error measured for depth cell. [DEPTH CELL Z + BIAS = TRUE Z]
Attributes
Modification

Position

Definition — reset vehicle dead reckon position to (x, y) or (x, y, z) = (#, ##, ###) at current clock time This is a navigational position fix. Receipt of a POSITION/LOCATION/FIX command resets the execution level dead-reckon position. Note that depth value z will likely be reset by depth cell if operational. During virtual world operation, hydrodynamics model is rezeroed.
Attributes
Modifications

Orientation

Definition — reset vehicle orientation to (phi, theta, psi) = (#, ##, ###) During virtual world operation, hydrodynamics model is rezeroed.
Attributes
Modifications

Posture

Definition — reset vehicle dead reckon posture to (x, y, z, phi, theta, psi) = (#a, #b, #c, #d, #e, #f)
Attributes
Modifications

OceanCurrent

Definition — Ocean current rate along North-axis, East-axis and [optional] Depth-axis (feet/sec) (this is cartesian version of parametric set and drift)
Attributes
Modifications

SeaState

Definition — Estimate of surface sea state, rounds to integer [0..9]
Attributes
Modifications

Waypoint

Definition — Point towards waypoint with coordinates (#X, #Y) (depth #Z optional) (speed #rpm optional). You can leave waypoint control by ordering course, rudder, sliding-mode, rotate or lateral thruster control. If speed is < 200 RPM, port & starboard RPMs are increased to 400 RPM to ensure waypoint can be achieved. If in TACTICAL mode, execution reports STABLE when waypoint is achieved.
Attributes
Modifications

StandoffDistance

Definition — Change standoff distance for WAYPOINT and HOVER control. Default value is 2.5 feet for NPS AUV, 50.0 feet for SSN. Default values are automatically read from control.constants.input.hulltype files.
Attributes
Modifications

Hover

Definition — Hover at present position and ordered depth using thrusters and propellers. HOVER without parameters is the preferred method of slowing since backing down with negative propellers may result in large sternway and severe depth excursions. [#X #Y] [#Z] Hover using thrusters and propellers for lateral and longitudinal positioning at specified position. Default Z value is previously ordered DEPTH.

Attributes
Modifications Needs attribute to enable/disble mode

TargetStation
Definition Hover relative to a sonar target at range = #R and target bearing #B from the AUV. Commanded AUV heading is #Psi (default is point at target). Stationkeeping will use full target tracking sonar mode
Attributes
Modification

TargetPoint
Definition Turn off stationkeeping control mode
Attribute
Modifications

EnterTube
Definition Experimental control mode. This tells execution level that nose has entered the tube, drive the rest of the way in using dead reckon for forward motion and sonars (pointing to opposite sides) to maintain tube side wall standoff. Parameters: How far forward to travel to be fully inside tube Tube orientation in degrees
Attributes
Modifications

// Mission timing commands ----------//

Wait
Definition Wait (or run) for # seconds (letting the robot execute) prior to reading from the script file again)
Attribute
Modifications

WaitUntil
Definition Wait (or run) until robot clock reaches time #a (letting the AUV execute its current orders) prior to reading from the script file again. If #a is earlier than current time, reset the clock. If in TACTICAL mode, command is ignored.
Attributes
Modifications

TimeStep
Definition change default execution level time step interval from default of 0.1 sec to # sec
Attributes
Modification

SingleStep
Definition Only useful in execution keyboard mode.
Attributes
Modification

Pause
Definition temporarily stop execution until <enter> is pressed
Attribute
Modification

RealTime
Definition run execution level code in real-time (busy wait at the end of each timestep if time remains)
Attribute
Modification

// Mission setup and configuration commands ----------//

LocationLab
Definition Vehicle is operating in lab using virtual world (default mode)
Attribute

Modification

Tethered

Definition command line switch only, used for in-water runs

Attribute

Modification

VirtualHost

Definition which is already running and waiting on 'hostname'

Attribute

Modification

Mission

Definition Replace temporary file 'mission.script' with 'filename'

Attribute

Modification

Telmetry

Definition Playback prerecorded telemetry data from filename.

Attribute

Modification

NoScript

Definition Ignore script command file. Selectively used in combination with TELEMETRY data file playback.

Attribute

Modification

Keyboard

Definition read script commands from keyboard

Attribute

Modification

Trace

Definition enable verbose print statements in execution level

Attribute

Modification

LoopForever

Definition repeat current mission when done, indefinitely.

Attribute

Modification

ControlConstantSFilename

Definition read revised control coefficients from "filename"(i.e. control.constants.input.auv, ..ssn, suboff and overwrite default file control.constants.input).

Attributes

Modification

Text

Definition Turn text display in command window back on

Attribute

Modification

Exit

Definition do not execute any more commands in this script, but repeat the mission again if LOOP-FOREVER is set

Attribute

Modification

// Sonar commands ----------//

Sonar725

Definition Set the bearing (#b), range (#r), and power (#p) of the ST725 sonar. In virtual world, bearing is necessary for sonar model. In water, this stores data in the execution level state vector for replay and examination. ST725 is electronically controlled by the tactical level laptop.Optional [#d] direction: TRUE or RELATIVE

Attributes
Modifications

Sonar1000
Definition Manually control the 000 sonar bearing to #b degrees relative to Phoenix heading. ST1000 is electronically controlled by the execution level Gespac serial port.Optional [#d] direction: TRUE or RELATIVE
Attribute
Modification

ST1000ScanWidth
Definition Total degrees for default sweep sonar scan [#], centered about bow
Attribute
Modification

SonarTrace
Definition Enable verbose print statements in execution sonar code
Attribute
Modification

// Miscellaneous commands ----------//

Sound
Definition enable text-to-speech audio output
Attribute
Modifications

Email
Definition ask user for electronic mail address at mission start, send user an electronic mail report at mission finish
Attribute
Modification

SlidingModeCourse
Definition Sliding mode course control algorithm (not yet working)
Attribute
Modifications

ParallelPortTrace
Definition enable trace statements for parallel port communications
Attribute
Modifications

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX G – CNO INTERVIEW: NPS OFFERS INNOVATION AND ASYMMETRIC ADVANTAGE

CNO: NPS Offers Innovation and Asymmetric Advantage
Story Number: NNS030515-02
Release Date: 5/15/2003 9:57:00 AM
http://www.news.navy.mil/search/display.asp?story_id=7466

By Journalist 2nd Class J. Anthony Reese, Naval Postgraduate School Public Affairs

MONTEREY, Calif. (NNS) -- "The difference between NPS and other universities is that the students get an invaluable opportunity at an education dealing with the issues of the Navy that cannot be achieved at any other institution," Chief of Naval Operations Adm. Vern Clark said May 14 at the Naval Postgraduate School (NPS).

The Chief of Naval Operations received briefs on NPS research in ship systems design and functionality. Also addressed were autonomous unmanned vehicles using undersea network nodes and autonomous behaviors and 3-D visual simulations for fleet use in anti-terrorist/force protection programs, according to Dean of Research Professor Dave Netzer.

Clark learned about an expeditionary warfare design and development project involving 92 students and 18 faculty members from seven academic programs, noted Professor Charles Calvano, Wayne E. Meyer Institute of Systems Engineering.

"The Navy is firmly committed to the growth and development of its personnel," Clark said. "When I talk to audiences around the world, I say that our asymmetric advantage starts with the education of our people. So when we're talking about NPS as a corporate university we're talking about the centerpiece of developing that genius."

During the brief on autonomous vehicles and a discussion about data transfer from unmanned platforms to submarines, Clark queried Navy doctoral candidate Capt. John Nicholson about the focus of his research and the needs of the Navy. "How do we come to grips with these technical issues and push this back to the fleet quickly?"

"We must challenge the paradigms of current long-distance, underwater communication methods," said mechanical engineering Professor Tony Healey. "We can tackle this issue and be credible and innovative because we have real experimental vehicles and understand the needs of today's Navy."

Clark got a firsthand look at a simulation model created by NPS students to test anti-terrorism decision-making skills. He then challenged the students to accelerate the transfer of knowledge from student thesis research to fleet and military operations.

"You'll never get a closer linked education to our profession than you'll get here at NPS," he said.

For related news, visit the Naval Postgraduate School Navy NewsStand page at www.news.navy.mil/local/nps.

LIST OF REFERENCES

AN/SLQ-48 Mine Neutralization System http://www.ae.utexas.edu/~industry/mine/mns.html Accessed June 2003

AN/WLD-1 RMS Remote Minehunting System http://www.fas.org/man/dod-101/sys/ship/weaps/rms.htm Accessed June 2003

ARTS/IXRetail XML Event at NRF Annual Convention http://www.nrf-arts.org/ppt/nrfshow/artsxml.htm June 2003

Berners-Lee, Tim, Hendler, James and Lassila, Ora. The Semantic Web, Scientific American, May 2001

Brailsford David F., Just what is XML? http://users.eggconnect.net/bcs.nottmderby/images/bcsxml01.pdf accessed June2003

Commerce Business Daily Issue June 21, 2001. Remote Environmental Monitoring Unit Autonomous Underwater Vehicles. http://www.fbodaily.com/cbd/archive/2001/06(June)/21-Jun-2001/58sol002.htm Accessed June 2003

Defense Technical Information Center. http://www.dtic.mil/doctrine/jel/doddict/data/i/02749.html Accessed March 2003.

Deitel, H. M., Deitel, P. J., Nieto, T. R., Lin, T. M. and Sadhu, P. XML: How To Program. Prentice Hall, Inc. Upper Saddle River, New Jersey, 2001

Digital Signature Activity Statement, W3C www.w3.org/Signature/Activity.html June 2003.

DoD dictionary of Military Terms http://www.dtic.mil/doctrine/jel/doddict/data/i/02749.html Accessed May 2003

DoD Metadata Registry and Clearing House http://diides.ncr.disa.mil/xmlreg/user/index.cfm Accessed June 2003.

DoD XML Registry v3.1.0.4 http://diides.ncr.disa.mil/xmlreg/user/index.cfm April 2003

EDN Access – *Underwater modem meets the challenge of a difficult channel – but slowly* by Bill Schweber 1/4/2001 http://www.e-insite.net/ednmag/index.asp?layout=article&articleid=CA60930 Accessed June 2003.

GCCS – Global Command and Control System – United States Nuclear Forces. www.fas.org/nuke/guide/usa/c3i/gccs.htm Accessed June 2003

Global Command and Control System-Maritime (GCCS – M) AN/USQ-119E(V www.fas.org/man/dod-101/sys/ship/weaps/gccs-m.htm Accessed June 2003

Gruneisen, Adrien and Henriet, Yann, 3D Model of ARIES Autonomous Underwater Vehicle (AUV) JavaDoc for Dynamics, Software, AUV Mission-Visualization Workbench, and AUV Dynamics Control Workbench in Mathlab October 2002

Hollander, Dave and Sperberg-McQueen, C. M., "Happy Birthday, XML!", 2003. Available at: http://www.w3.org/2003/02/xml-at-5.html

Intro to XSL http://www.w3schools.com/xsl/xsl_intro.asp Accessed April 2003

Introducing XML Serialization http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconintroducingxmlserialization.asp Accessed June 2003.

Introduction to XML Schema http://www.w3schools.com/schema/schema_intro.asp Accessed April 2003

Ipedo Web http://www.ipedo.com/downloads/products_high_performance_XML.pdf June 2003

Long Term Mine Reconnaissance System (LMRS) http://www.fas.org/man/dod-101/sys/ship/weaps/lmrs.htm Updated Friday January 19, 1999. Accessed June 2003

Long Term Mine Reconnaissance System Statement of Performance and Results for the Detailed Design Phase http://www.fas.org/man/dod-101/sys/ship/weaps/docs/lmrs_spr.html 4 August 1997. Accessed June 2003.

Marshall, William J., III and Lehr, Steven E., Mine Warfare, An Enduring Challenge. National Defense Industry Association, November 1998

May 1999 Hl7-XML Progress Report http://www.infoloom.com/gcaconfs/WEB/granada99/als.HTM June 2003

Mine Warfare History -http://www.exwar.org/1800_history/mine/mine.htm Accessed February 2003)

Mine Warfare. NWP 3-15. Department of the Navy. August 1999

Module 8 – Intelligence Automated Data Processing (ADP)Systems www.fas.org/irp/doddir/navy/rfs/part08.htm Accessed June 2003

MSXML Pages, "Valid XML", 2003. Available at: http://msxml.com/xml_tutorial/valid-xml.html

Naval Mine History [AMCM] (http://members.aol.com/helmineron/minehist.htm) June 2003

Naval Coastal Sea Systems http://www.ncsc.navy.mil/contracts/miller%2C%20j/03r0043.htm Accessed March 2003

Navy Fact File: Naval Mines http://www.chinfo.navy.mil/navpalib/factfile/weapons/wep-mine.html Accessed June 2003

Naval Warfare Development Center: Fleet Battle Experiment Juliet. *New Technology Helps Shape Future of Navy's Forces* JO2 Stacie Rose July 2002. http://www.nwdc.navy.mil/Conference/FBEJ/7-28-release.asp Accessed June 2003

NPS/CIRPAS Activity Statement. 2001 http://web.nps.navy.mil/~cirpas/Projects/KB01%20Activity%20Summary.pdf Accessed June 2003

Office of Naval Research: BPAUV Battlespace Preparation Autonomous Underwater Vehicle. http://www.onr.navy.mil/sci_tech/ocean/docs/bpauv.pdf Accessed June 2003.

Operational Requirements Document for the Long Term Mine Reconnaissance System May 2, 1996 http://www.fas.org/man/dod-101/sys/ship/weaps/docs/lmrs_ord.html Accessed June 2003.

Reese, Anthony *CNO: NPS Offers Innovation and Asymmetric Advantage. 5/15/2003.* Naval Postgraduate School Public Affairs.

Reimers, Stephen Paul. Towards Internet Protocol Over Seawater (IP/SW): Forward Error Correction (FEC) Using Hamming Codes for Reliable Acoustic Telemetry. September 1995.

RMS Brochure http://www.lockheedmartin.com/syracuse/ocean/mine_neut/RMSbrochure.pdf Accessed June 2003

Schools Interoperabiltiy Framework http://www.sifinfo.org/press_110901.html June 2003

Scientific American *The Semantic Web – A new form of Web content that is meaningful to compute*. May 2001. http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21&pageNumber=2&catID=2 Accessed June 2003

Serin, Ekrem, "Design and Test of the Cross-Format Schema Protocol (XFSP) for Networked Virtual Environments", 2003. Available at: http://theses.nps.navy.mil/03mar_serin.pdf

Semantic Web http://www.w3.org/2001/sw/ Accessed May 2003

The Bushnell Keg Mine http://www.ae.utexas.edu/~industry/mine/bushnell.html Accessed June 2003

The Cover Pages: New OTA XML Specification http://xml.coverpages.org/OpenTravelAllianceCustomerProfile20010524.html June 2003

The Extensible Stylesheet Language (XSL) W3C http://www.w3c.org/Style/XSL Accessed April 2003.

The Net Gets Wet: The Navy Announces It Has Finally Conquered One Of The Toughest Internet Frontiers: The Ocean. Mark Schrope.
http://www.business2.com/articles/mag/print/0,1643,14163,FF.html Accessed June 2003

USS AVENGER MCM1 http://www.avenger.navy.mil/ Accessed June 2003

UUV Master Plan: A Vision for Navy UUV Development. Presented by Barbara Fletcher; Space and Naval Warfare Systems Center, San Diego, CA.
http://www.spawar.navy.mil/robots/pubs/oceans2000b.pdf Accessed June 2003

Valid XML http://msxml.com/xml_tutorial/valid-xml.html Accessed April 2003

W3C Semantic Web http://www.w3.org/2001/sw/ Accessed June 2003

Walsh, N. "What is XML?", 3 October 1998. Available at:
http://www.xml.com/pub/a/98/10/guide1.html - AEN58

Web Year.
http://searchwebservices.techtarget.com/gDefinition/0,294236,sid26_gci853845,00.html
Accessed June 2003

Weekley, Jeffrey D. Information Processing for AUV Operations using an Open Source Approach. Naval Postgraduate School, 2003.

Welcome to Hydroid, Inc. – Home of the REMUS AUV. www.hydroidinc.com Accessed June 2003

Welcome to IMS Global Learning Consortium, Inc. www.imsproject.org Accessed June 2003.

What is the Joint Command &Control System (GCCS-J) http://gccs.disa.mil/gccs Accessed June 2003.

What is XSL? http://www.w3.org/Style/XSL/WhatIsXSL.html Accessed April 2003

Whitman, Edward C. Unmanned Underwater Vehicles: Beneath the Wave of the Future. 2002.

WHOI at Sea Remote Environmental Monitoring Units
http://www.whoi.edu/home/marine/remus_main.html Accessed June 2003

World Wide Web Consotrium (W3C) www.w3.org/Consortium accessed April 2003
World Wide Web Consortium (W3C), "About the World Wide Web Consortium (W3C)", 2003. Available at: http://www.w3.org/Consortium/

World Wide Web Consortium (W3C), "The Extensible Stylesheet Language (XSL)" 2003. Available at: http://www.w3c.org/Style/XSL

Wrox Diagram http://www.perfectxml.com/Conf/Wrox/Files/brianl_xml.pdf June 2003

XML eXtensible Markup Language: *An Introduction*
http://xml.gov/presentations/gsa/marion_royal_intro_to_xml.PPT June 2003

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Daniel J. Stilwell
   The Bradley Department of Electrical & Computer Engineering
   Virginia Polytechnic Institute & State University
   Blacksburg, Virginia

4. RADM (ret.) John Pearson, USN
   Mine Warfare Chair
   Naval Postgraduate School
   Monterey, California

5. Dr. Donald Brutzman, Code UW/Br
   Undersea Warfare AcademicGroup
   Naval Postgraduate School
   Monterey, California

6. Professor Anthony Healey, Code ME/HY
   Department of Mechanical Engineering
   Naval Postgraduate School
   Monterey, California

7. Dr. T. Swean, Code 320E
   Office of Naval Research
   Arlington, Virginia

8. Doug Horner
   Naval Postgraduate School
   Monterey, California

9. Jeff Weekley
   Naval Postgraduate School
   Monterey, California

10. CAPT Jeff Kline
    Chair, Warfare Innovation
    Naval Postgraduate School
    Monterey, California

11. Dr. Dan Boger
Chair, Information Sciences
Naval Postgraduate School
Monterey, California

12. Capt Darrin Hawkins
Air Force Communications Agency
Scott Air Force Base, Illinois

13. ENS Barbara Van Leuvan
6262 Emerson Ave S #28
St. Petersburg, FL 33707